

ESTRUCTURA DE COMPUTADORES

PRIMER PARCIAL (2 de Noviembre de 2010)

1 (1 punto) Indique, justificando su respuesta, si las siguientes afirmaciones son verdaderas o falsas:

a) La arquitectura von Neumann en la que se basan los computadores tradicionales consiste en tener almacenados los datos separados de las instrucciones en memorias distintas.

Falso. Se caracteriza por tener en una memoria almacenados tanto instrucciones como datos.

a) El registro de estado es un registro transparente al usuario, ya que éste no tiene por qué utilizarlo en las instrucciones máquina.

Falso. Es un registro de propósito específico. El usuario puede ejecutar por ejemplo instrucciones de salto condicional para ver algunas condiciones.

a) La unidad de control necesita como entrada el registro contador de programa, para saber cuál es la instrucción que debe ejecutar a continuación.

Falso. La unidad de control necesita conocer la instrucción, que se encuentra en el registro de instrucción.

a) El registro de direcciones de memoria es un registro de propósito general que puede contener tanto direcciones como datos.

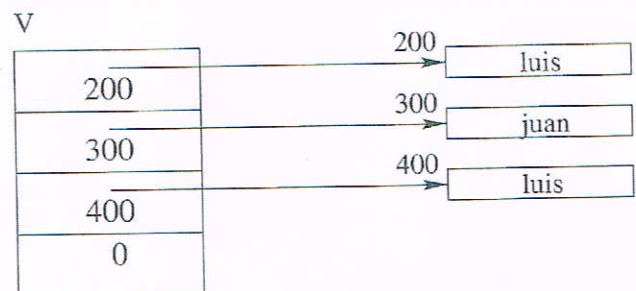
Falso. Es un registro transparente al usuario, que tan sólo puede tener direcciones.

2 (2,5 puntos) Programe en ensamblador del 88110 la función mapa que recibe dos parámetros en la pila:

- *v*: Es un vector que contiene direcciones de comienzo de cadenas de caracteres. La dirección 0 indicará el final del vector cuyo tamaño nunca será superior a 32 elementos. Se pasa por dirección.
- *str*: Es una cadena de caracteres que se pasa por dirección.

Esta función devolverá en r29 una máscara que contendrá en el bit *i*-ésimo un 1 si la cadena *str* es igual a la cadena apuntada por el elemento *i*-ésimo del vector o 0 en caso contrario. Se recuerda que el formato de la instrucción para poner un campo de bits a 1 es `set rD,rS1,W5<05>` o `set rD,rS1,rS2`. En este último caso los bits 9-5 y 4-0 de rS2 se toman como los campos W5 (ancho del campo de bit) y 05 (bit origen del campo de bit) respectivamente.

Se supondrá que existe la función de librería `strcmp(str1,str2)` que compara las cadenas de caracteres *str1* y *str2* y devuelve un 0 en r29 si son iguales o 1 si no lo son. El resultado de invocar a la función `mapa` cuando el parámetro *str* contiene el valor *luis* y *V* es el vector que aparece en la figura es: `r29 = 0x00000005`.



SOLUCIÓN

Para recorrer el vector se crearán tres variables locales en el marco de pila de la subrutina: el índice del bit que hay que poner a 1 en caso de que la entrada del vector sea igual a *str* (posición -12); la máscara que será el resultado de la subrutina (posición -8); y la dirección en curso del vector *v* (posición -4). La función recorre el vector llamando a `strcmp` con la entrada en curso del vector y el parámetro *str*. Previamente a la llamada se habrán salvaguardado las variables locales en sus lugares correspondientes del marco de pila y después de la llamada se recuperan de la pila. Si son iguales se pone a 1 el bit indicado en los cinco bits menos significativos de r4.

```

mapa:  PUSH(r1)
        PUSH(r31)
        subu r30,r30,12      ; Se reservan tres variables locales:
                                ; -12: bit que hay que poner a 1
                                ; -8: máscara resultado
                                ; -4: dirección en curso del vector v

        or r3,r0,r0
        ld r20,r31,8
        or r4,r0,0x20        ; Se inicializa W5 a 1 y O5 a 0
bucle:  st r4,r31,-12
        st r3,r31,-8
        st r20,r31,-4
        ld r21,r20,r0        ; Si la entrada es 0 se finaliza.
        cmp r5,r21,r0
        bbl eq,r5,fin
        PUSH(r21)            ; Se pasa la entrada del vector
        ld r21,r31,12        ; Se pasa str
        PUSH(r21)
        bsr strcmp
        addu r30,r30,8        ; Se eliminan los parámetros
        ld r3,r31,-8          ; Se recupera la máscara resultado,
        ld r20,r31,-4         ; el puntero en curso a v y
        ld r4,r31,-12         ; el bit que hay que poner a 1
        cmp r5,r29,r0
        bbl eq,r5,no_iguales
        set r3,r3,r4          ; Se pone a 1 el bit que corresponde
no_iguales: addu r4,r4,1      ; Siguiente bit
        addu r20,r20,4        ; Siguiente elemento del vector
        br bucle
fin:    or r29,r3,r3
        or r30,r31,r31
        POP(r31)
        POP(r1)
        jmp(r1)

```

3 (2 puntos) Programe en ensamblador del 88110 los fragmentos de la subrutina SUBROUTINA que se detallan a continuación:

```

SUBROUTINA (int p1, int p2, int p3)
{
    int vl_a = 0;    /* vl_a = 0 */
    char vl_b = 0;   /* vl_b = null 8 bits */
    char vl_c = 45;  /* vl_c = 'A' 8 bits */

    /* Fragmento 1 */
    ....
    /* Fragmento 2 */

    funcion (0, p2, vl_a);

    /* Fragmento 3 */
    ....
    /* Fin */
}

```

- Marco de pila: Creación del marco de pila incluyendo la inicialización de las variables locales.
- Fragmento 2: Llamada a la subrutina funcion incluyendo el paso de los tres parámetros por valor y en la pila. Suponga que en el Fragmento 2 se hace uso de los registros r5, r6, r7, r8, r9 y r10; que r9 y r10

contienen información relevante para el Fragmento 3; y que la subrutina llamante, SUBROUTINA, deja disponibles todos los registros excepto r1, r30 (SP) y r31 (FP).

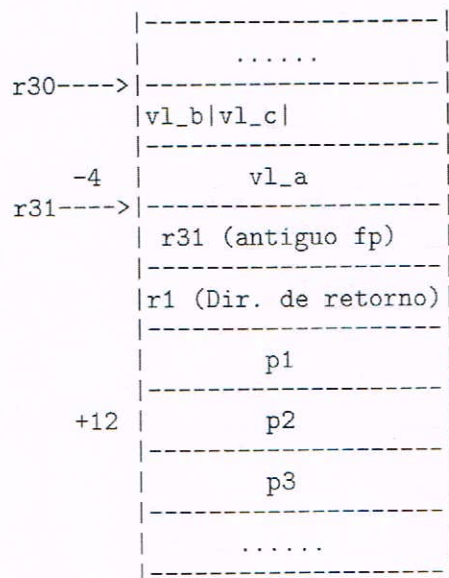
c) Fin: Retorno al programa llamante, incluyendo la destrucción del marco de pila.

SOLUCIÓN

a) Fragmento 1. Creación del marco de pila:

```
SUBROUTINA:  PUSH (r1)           ; Salva en la pila la dirección de retorno
              PUSH (r31)        ; Salva en la pila el puntero de marco de pila
              or r31, r30, r30   ; Crea puntero de marco de pila
              subu r30, r30, 8    ; Crea espacio en la pila para las variables locales
              st r0, r31, -4      ; Inicializa vl_a=0
              st.b r0, r31, -5    ; Inicializa vl_b=0
              or r2, r0, 45       ; 
              st.b r2, r31, -6    ; Inicializa vl_c=45
```

Estado en que queda la pila:



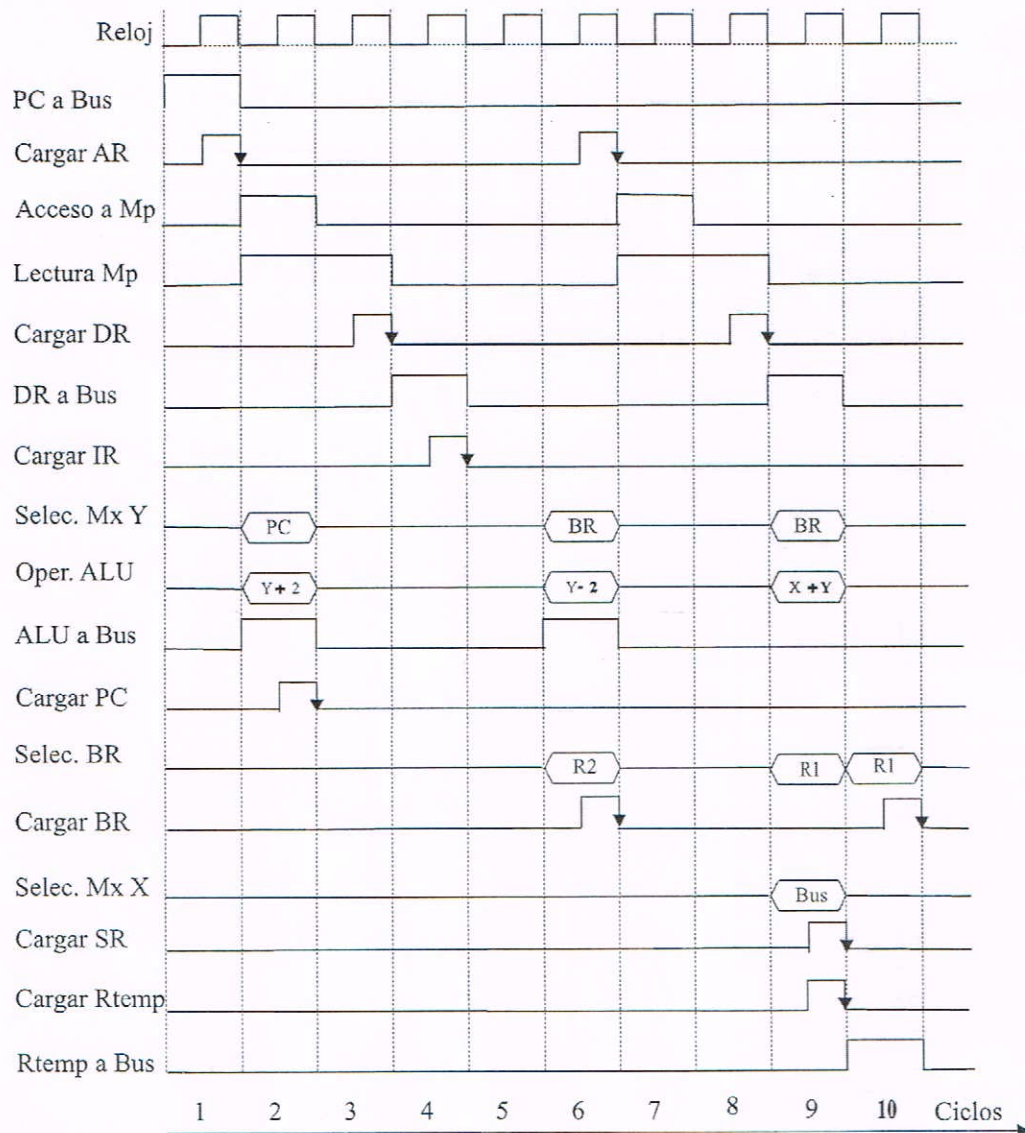
b) Fragmento 2. Llamada a la subrutina funcion:

```
PUSH (r9)           ; Salva registro r9
PUSH (r10)          ; Salva registro r10
ld r2, r31, -4      ; r2=vl_a
PUSH (r2)           ; 3er parámetro: vl_a
ld r2, r31, 12      ; r2=p2
PUSH (r2)           ; 2º parámetro: p2
PUSH (r0)           ; 1er parámetro: 0
bsr FUNCION         ; llamada a la subrutina
```

c) Fragmento 3. Retorno:

```
or r30, r31, r31    ; Destrucción de variables locales
POP (r31)           ; Recuperación del puntero de marco de pila
POP (r1)            ; Recuperación de la dirección de retorno
jmp (r1)            ; Retorno
```

4 (2 puntos) En la figura adjunta se muestra el cronograma de las señales de control que se activan en un computador con palabras y direcciones de 16 bits y direccionamiento a nivel de byte, durante la ejecución de una instrucción de las 250 de que consta su repertorio de instrucciones. La Unidad de control de este computador es cableada y el Banco de Registros (BR) contiene 15 registros de propósito general (R0 a R14).



a) Represente a nivel RT (transferencia entre registros) las operaciones elementales que se producen en cada uno de los diez ciclos de esta ejecución.

b) Explique qué ciclos corresponden a la fase de fetch y cuales a la fase de ejecución. ¿Por qué en el ciclo 5 no se activa ninguna de las señales de control?

c) A partir de la representación del apartado a), determine de qué instrucción se trata y qué operaciones realiza. Indique su formato, sus operandos y los modos de direccionamiento de estos.

SOLUCIÓN

a) Las operaciones elementales, correspondientes a los ciclos representados en el cronograma, son las siguientes:


```

1:  AR ← PC
2:  PC ← PC+2
   Comienza ciclo de lectura
3:  DR ← M(AR)
4:  IR ← DR
5:  No se activan señales de control
6:  AR, R2 ← R2-2
7:  Comienza ciclo de lectura
8:  DR ← M(AR)
9:  Rtemp ← R1+DR
   Actualizar SR
10: R1 ← Rtemp

```

b) La fase de *fetch* de la instrucción se corresponde con los cuatro primeros ciclos del cronograma y la fase de ejecución con los ciclos 6 a 10. En el ciclo 5 no se activan las señales de control porque éste se utiliza en la decodificación de la instrucción.

c) Esta instrucción decrementa el registro R2, que contiene una dirección de Memoria ya que se carga también el resultado en el registro AR. La lectura de Memoria obtiene el dato, que se suma con el contenido del registro R1, actualizando el registro de estado. Finalmente, el resultado de la suma se carga en el registro R1. Se trata, por lo tanto, de una instrucción aritmética que suma los contenidos de un registro y de una dirección de Memoria, con predecremento, cargando el resultado en el registro. La instrucción, sus operados y direccionamientos son:

ADD .R1, [--.R2]

Es una instrucción de una palabra, ya que no se incrementa el PC en la fase de ejecución. Puesto que las palabras son de 16 bits, el juego de instrucciones consta de 250 instrucciones y el Banco de Registros contiene 15 registros, su formato es:

C.O.	Registro	Registro
8	4	4

5 (2,5 puntos) Sea un computador cuyo tiempo de ciclo es de 15 ut y el tiempo de acceso a memoria es de 40 ut. Micro programe a nivel RT la instrucción de dos palabras que se indica a continuación, indicando qué microinstrucciones pertenecen a cada una de las fases de ejecución de la instrucción: CALL /1000, e incluyendo el fetch de la siguiente instrucción. Suponga que la pila crece en direcciones crecientes de memoria y el puntero de pila apunta a la última palabra introducida. Indique el tiempo total de ejecución de dicha instrucción.

SOLUCIÓN

El microprograma de la instrucción CALL /1000, procurando solapar operaciones elementales puede ser el siguiente:

```

. Decodificación
. AR ← PC ; leer dirección (segunda palabra)
. DR ← M(AR), PC++
. TMP1 ← DR ; guardar en registro temporal la dirección de salto
. AR, SP ← SP++ ; preincrementar SP
. DR ← SP
. DR → M(AR); guardar en pila la dirección de retorno
. PC, AR ← TMP1; saltar y fetch (solapado un ciclo)
. DR ← M(AR), PC++
. RI ← DR, ir a CO

```

La memoria necesitaría 3 ciclos para operar, ya que su tiempo de acceso es 40 ut y el del ciclo de reloj es 15 ut.

$$Teje = (10 + 3 \cdot 2) \text{ ciclos} = 16 \text{ ciclos} \cdot 15 \text{ ut/ciclo} = 240 \text{ ut}$$

ESTRUCTURA DE COMPUTADORES
RECUPERACIÓN DEL PRIMER PARCIAL (17 de Enero de 2011)

--	--	--

Apellidos, Nombre..... N° de Matrícula.....

1 (5 puntos) Sea un computador cuya memoria es asíncrona y activa una señal WAIT mientras no haya terminado la operación solicitada.

a) Desglose en operaciones elementales a nivel RT la instrucción de dos palabras: PUSH /dir, procurando solapar operaciones elementales, e incluyendo el fetch de la siguiente instrucción. La pila crece en direcciones decrecientes y el SP apunta al último dato introducido en ella.

La operación PUSH requiere un predecremento del SP.

```
. AR<- PC ; TMP<- PC+1 leer dirección (segunda palabra)
. DR<- M(AR), PC<- TMP Si WAIT ir a $;
. AR<- DR;
. DR<- M(AR); Si WAIT ir a $; leer dato a introducir en pila
. SP, AR<- SP-1; predecrementar SP
. M(AR)<- DR; Si WAIT ir a $; Escribir en pila
. ir a fetch
```

```
fetch: AR<- PC
. AR<- PC ; TMP<- PC+1
. DR<- M(AR), PC<- TMP Si WAIT ir a $;
. RI<- DR; ir a CO
```

b) Indique el tiempo necesario para un ciclo de reloj suponiendo que la unidad de control sea microprogramada, teniendo en cuenta que la operación elemental más larga dura 18 ut. y los retardos de algunos dispositivos son:

- | | |
|-----------------------------|--------------------------------------|
| ▪ Multiplexores: 1 ut | ▪ Acceso a Memoria de Control: 13 ut |
| ▪ Secuenciador de UC: 10 ut | ▪ Decodificador: 3 ut |

Para UC microprogramada, el tiempo de ciclo sería: $T_{ck} = T_r + T_{dec} + T_{op} + T_{sec} + 2 \cdot T_r + T_{mc} + T_r = 3 + 18 + 10 + 13 = 44 \text{ ut}$

Se supone despreciable el tiempo de acceso a un registro.

c) ¿Cuál sería el tiempo medio de ejecución de esta instrucción si la memoria principal en promedio necesita 2 ciclos de reloj?

$$T_{eje} = 10 + 4 \text{ ciclos} = 14 \text{ ciclos} \cdot 44 \text{ ut/ciclo} = 616 \text{ ut}$$

2 (5 puntos) Programe en ensamblador del 88110 la función **descompacta** que recibe cuatro parámetros en la pila:

- **v:** Es una matriz de enteros de m filas por n columnas almacenada por filas. Se pasa por dirección.
- **p:** Es una lista compacta que contiene elementos de una matriz. Se pasa por dirección. Cada uno de los elementos de la lista se compone de tres campos enteros:
 - **elemento:** Contiene el elemento que pertenece a la matriz.
 - **fila:** Indica la fila a la que pertenece el elemento. Las filas comienzan a numerarse en la posición 0.
 - **columna:** Indica la columna a la que pertenece el elemento. Las columnas comienzan a numerarse en la posición 0.

El final de la lista está indicado con el el valor -1 contenido en el campo fila.

- **m:** Es un entero que contiene el número de filas de la matriz v. Se pasa por valor.
- **n:** Es un entero que contiene el número de columnas de la matriz v. Se pasa por valor.

Esta función rellenará la matriz apuntada por v con los elementos que se pasan en la lista p y el resto de los elementos de la matriz serán nulos. Suponga que los datos que se pasan en la lista p son correctos, es decir, los campos de filas y columnas están dentro del rango de los parámetros m y n.

El resultado del ejemplo que se muestra en la figura será una matriz en la que su elemento (0, 5) contendrá el valor 15, el (1, 5) el valor 20 y el (4, 8) el valor 40. El resto de los elementos de la matriz serán 0.

P	20	1	5	15	0	5	40	4	8	5	-1	-1
---	----	---	---	----	---	---	----	---	---	---	----	----

SOLUCIÓN

```

descompacta: ld r20,r30,0      ; Se carga v
              ld r21,r30,4      ; Se carga p
              ld r4,r30,8       ; Se carga m
              ld r5,r30,12      ; Se carga n
              mulu r4,r4,r5
              or r7,r0,r0
b_i:          st r0,r20,r7      ; Se inicializa la matriz
              addu r7,r7,4      ; resultado con todos sus
              subu r4,r4,1      ; elementos a 0
              cmp r6,r4,0
              bbl ne, r6, b_i
bucle:        ld r6,r21,0       ; Se carga elem
              ld r7,r21,4       ; Se carga fila
              cmp r9,r7,-1      ; Si es -1 se acaba
              bbl eq, r9, fin
              ld r8,r21,8       ; Se carga columna
              mulu r7,r7,r5      ; Se calcula la dirección
              addu r7,r7,r8      ; del elemento
              mak r7,r7,0<2>     ; de la matriz
              st r6,r20,r7      ; Se almacena el elemento
              addu r21,r21,12    ; Se pasa al siguiente
              br bucle
fin:          jmp (r1)

```

ESTRUCTURA DE COMPUTADORES
SEGUNDO PARCIAL (17 de enero de 2011)

--	--

--

Apellidos, Nombre..... N° de Matrícula.....

1 Sea una unidad de disco duro con las siguientes características:

- 10 superficies con 3.000 pistas cada una (0-2.999).
- Cada pista tiene 160 sectores (0-159).
- Cada sector consta de 1.024 bytes de información neta y 1.280 de información bruta.
- Velocidad de transferencia $25,6 \cdot 10^6$ bytes/s.
- El tiempo de mover la cabeza de una pista a otra consecutiva es $10 \mu s$
- El tiempo de estabilización de las cabezas es de 2 ms.

a) (1 punto) Calcule el tiempo medio de acceso de esta unidad.

$$t_{lat} = \frac{\text{Información por pista}}{\text{Velocidad de transferencia}} = \frac{1.280 \text{ bytes/sector} \cdot 160 \text{ sectores/pista}}{25,6 \cdot 10^6 \text{ bytes/s}} = 8 \text{ ms}$$

$$\bar{t}_{búsq} = \bar{t}_{pos} + t_{est} = 0,01 \cdot \bar{n} \text{ ms} + 2 \text{ ms} = 0,01 \cdot 1.500 \text{ ms} + 2 \text{ ms} = 17 \text{ ms}$$

$$\bar{t}_{acc} = \bar{t}_{búsq} + \bar{t}_{lat} = 17 \text{ ms} + 4 \text{ ms} = 21 \text{ ms}$$

b) (1 punto) Determine el número de sector absoluto (lógico) correspondiente a las coordenadas geométricas (CHS) (1.234, 5, 67) y las coordenadas geométricas del sector absoluto n° 654.321.

$$(1.234, 5, 67) \rightarrow 1.234 \times 160 \times 10 + 5 \times 160 + 67 = 1.975.267$$

$$\begin{array}{rcl} \frac{654.321}{160 \cdot 10} & = & 408 \\ 654.321 \bmod 160 \cdot 10 & = & 1.521 \\ \frac{1.521}{160} & = & 9 \\ 1.521 \bmod 160 & = & 81 \\ 641.853 & \rightarrow & (408, 9, 81) \end{array}$$

c) (1 punto) Suponiendo que en el instante $t=0$ las cabezas de la unidad se encuentran al comienzo del sector 80 del cilindro 800, calcule en qué instante termina la operación de lectura del sector (1.600, 8, 100).

$$t_{búsq} = 0,01 \cdot n \text{ ms} + 2 \text{ ms} = 0,01 \cdot 800 \text{ ms} + 2 \text{ ms} = 10 \text{ ms}$$

En $t=10$ ms se encuentra en el cilindro 1.600 y como gira a razón de una revolución cada 8 ms, habrá girado 1,25 vueltas, es decir 200 sectores. De este modo las cabezas están al comienzo del sector $80 + 200 \bmod 160 = 120$.

Para alcanzar el sector 100, habrán de pasar $160 - 120 + 100 = 140$ sectores. Si en 8 ms pasan 160 sectores, en 7 ms se encuentra al comienzo del sector objetivo y 0,05 ms después lo habrá leído.

$$t_{acc} = 10 \text{ ms} + 7 \text{ ms} + 0,05 \text{ ms} = 17,05 \text{ ms}$$

2 Sea un computador con una CPU capaz de ejecutar 1.000 MIPS y con un sistema de interrupciones de varios niveles de prioridad que permite anidamiento e inhibición selectiva y cuya secuencia de reconocimiento de interrupciones (SRI) tiene una duración equivalente a 5 instrucciones. Este computador, con ancho de palabra de 32 bits, dispone de una unidad de disco duro con las siguientes características:

- Velocidad de transferencia: $16 \cdot 10^6$ bytes/s.
- Tiempo medio de acceso: 4 ms.
- Tamaño de bloque: 1.024 bytes.

El módulo de E/S de la unidad de disco opera mediante interrupciones con los siguientes parámetros:

- Buffer de 2 registros de datos de 32 bits.
- La rutina de inicio de operación consta de 200 instrucciones.
- La rutina de tratamiento de las interrupciones del disco duro consta de 45 instrucciones.
- La rutina de fin de operación consta de 100 instrucciones.

a) (1 punto) Calcule cuánto dura una operación de E/S para la lectura de un sector en este dispositivo.

$$t_{inst} = \frac{1}{10^9} \text{ s} = 1 \text{ ns}$$

$$\begin{aligned} t_{op} &= t_{ini} + t_{acc} + t_{trans} + t_{int} + t_{fin} = \\ &= 200t_{inst} + 4\text{ms} + \frac{1.024}{16 \cdot 10^6} \text{ s} + (5 + 45)t_{inst} + 100t_{inst} = \\ &= 200\text{ns} + 4\text{ms} + 64\mu\text{s} + 50\text{ns} + 100\text{ns} = 4.064.350\text{ns} \end{aligned}$$

b) (1 punto) Determine cuántas instrucciones puede ejecutar como máximo la CPU antes de atender una solicitud de interrupción de la unidad de disco. Indique qué sucedería en el caso de que la CPU tardara más tiempo en atender la solicitud de interrupción.

$$t_{entre_interrupciones} = \frac{2 \times 4 \text{ B}}{16 \cdot 10^6 \text{ B/s}} = 500\text{ns}$$

$$t_{int} = (5 + 45)t_{inst} = 50\text{ns}$$

$$t_{respuesta} = 500\text{ns} - 50\text{ns} \equiv 450 \text{ instrucciones}$$

Si la CPU tardara más tiempo en responder, el módulo de E/S no tendría espacio para almacenar los nuevos datos enviados por el disco: estos se perderían y fallaría la operación de E/S.

Se desea conectar varias unidades de disco duro a este computador para usarlo como servidor de discos.

c) (1 punto) Determine cuál es el número máximo de discos que pueden operar simultáneamente.

$$\text{Frecuencia de interrupciones } HD = \frac{16 \cdot 10^6 \text{ B/s}}{2 \times 4 \text{ B/interr}} = 2 \cdot 10^6 \text{ interr/s}$$

$$\text{Consumo de CPU } HD = 2 \cdot 10^6 \text{ interr/s} \times 50 \text{ instrucciones/interr} = 100 \cdot 10^6 \text{ instrucciones/s} = 100 \text{ MIPS}$$

$$\text{Número de discos} = \frac{1.000 \text{ MIPS}}{100 \text{ MIPS}} = 10$$

Este computador dispone también de una conexión a una red Ethernet con las siguientes características:

- Operación por acceso directo a memoria (DMA).
- Velocidad de transmisión de 1 Gigabit por segundo (10^9 bits/s).
- Buffer de 16 registros de datos de 32 bits.
- Las rutinas de inicio y fin de una operación de E/S para la transferencia de un bloque de datos constan de 200 y de 100 instrucciones respectivamente.
- La rutina de tratamiento de las interrupciones de la red Ethernet consta de 75 instrucciones.
- El protocolo de concesión y liberación de los buses dura 2 ns.
- El ciclo de acceso a memoria tiene una duración de 3 ns.

d) (2 puntos) Calcule el porcentaje de tiempo de CPU que se consume durante una operación de E/S para transmitir un bloque de 1.024 bytes usando la técnicas de DMA por ráfagas.

$$\begin{aligned} t_{op_eth_ráfaga} &= t_{ini} + t_{acc} + t_{dma} + t_{trans} + t_{int} + t_{fin} = \\ &= 200t_{inst} + 0 + (2ns + 16 \times 3ns) + \frac{1.024 \times 8}{10^9}s + (5 + 75)t_{inst} + 100t_{inst} = \\ &= 200ns + 0 + 50ns + 8.192ns + 80ns + 100ns = 8.622ns \end{aligned}$$

$$t_{cpu_eth_ráfaga} = t_{ini} + N_{dma} \times t_{dma} + t_{int} + t_{fin} = 200ns + 0 + \frac{1.024}{16 \times 4} \times 50ns + 80ns + 100ns = 1.180ns$$

$$\%_{cpu_ráfaga} = \frac{1.180ns}{8.622ns} \times 100 = 13,69 \%$$

En el computador se está ejecutando un programa de predicción meteorológica que no realiza ninguna operación de E/S y que emplea 1 millón de instrucciones en cada ciclo de simulación.

e) (2 puntos) Suponiendo que constantemente se encuentran operando 3 unidades de disco y también la red Ethernet, por la que se transmiten continuamente bloques de 1.024 bytes, determine cuántos pasos de simulación se ejecutan en un segundo (Desprecie el tiempo empleado por las rutinas de inicio y fin de operación de los discos para hacer este apartado.)

Cada disco consume un 10 % de la capacidad total de proceso (100 MIPS), de acuerdo con los cálculos realizados en el apartado c). Por su parte, la red Ethernet consume otro 13,69 %, según el apartado anterior, dando un total de $3 \times 10 + 13,69 = 43,69 \%$ que deja el 56,31 % para la ejecución del programa de simulación que, traducido a instrucciones por segundo, resulta en $1.000 \times 0,5631 = 563,1$ MIPS. Si cada paso de simulación consume 1 millón de instrucciones, en cada segundo se ejecutan 563,1 pasos.

1 Un computador cuenta con los dos formatos de representación siguientes :

- **Formato 1. Coma fija:** 16 bits en complemento a dos con la coma entre los bits 5 y 6, por ejemplo 0100011011,101001.
- **Formato 2. Coma flotante:** 16 bits, el bit superior es el signo, los seis siguientes el exponente, representado en exceso a 32 y los 9 siguientes la magnitud de la mantisa, que está representada en signo magnitud, con bit implícito y la coma a la derecha de éste.

a) Determine el rango y la resolución de ambos formatos.

b) Dadas las representaciones de coma flotante $A=H440B$ y $B=H'BDDA$, determine su valor decimal y represente ambos números en el formato de coma fija.

c) Realice paso a paso la operación $A+B$ en ambos formatos, dejando el resultado en el formato de partida. En el caso de coma flotante utilice dos bits de guarda y un bit retenedor, así como redondeo al más próximo.

d) Determine el error absoluto que se ha cometido en las operaciones anteriores.

SOLUCIÓN

a) Rango y resolución de los formatos.

Formato 1.

$$\begin{cases} 1000\ 0000\ 00,00\ 0000 & \rightarrow -2^{-9} \\ 0111\ 1111\ 11,11\ 1111 & \rightarrow 2^9 - 2^{-6} \end{cases}$$

$$\text{Rango} = [-2^{-9}, 2^9 - 2^{-6}]$$

$$\text{Resolución} = 2^{-6}$$

Formato 2.

Exponente: $[-32, 31]$. Como hay que reservar el exponente -32 para la representación del cero, el rango del exponente queda: $[-31, 31]$.

$$\text{Mantisa: } \pm \begin{cases} 1,000000000 & \rightarrow 1 \\ 1,111111111 & \rightarrow 2 - 2^{-9} \end{cases}$$

$$\text{Rango} = \pm [1 \cdot 2^{-31}, (2 - 2^{-9}) \cdot 2^{31}] \cup 0$$

$$\text{Resolución} = 2^{-9} \cdot 2^E$$

b) Valor decimal y representación en coma fija.

$$A = H'440B = 0\ 100010\ 000001011 = +1,000001011 \cdot 2^2 = 100,0001011 = 4,0859375$$

$$B = H'BDDA = 1\ 011110\ 111011010 = -1,111011010 \cdot 2^{-2} = -,011110110100 = -0,481445312$$

Cambio al formato de coma fija:

$$A = 0000000100,000101 = H'0105$$

$$B = -0000000000,011110 = 1111111111,100010 = H'FFE2$$

c) Suma $A+B$.

Formato 1.

$$\begin{array}{r} A \quad \quad \quad 0000000100,000101 \\ +B \quad \quad \quad 1111111111,100010 \\ \hline A+B \quad \quad \quad 0000000011,100111 \end{array}$$

$$A+B = 0000000011,100111 = H'00E7$$

Formato 2.

Los números a sumar son:

$$A = +1,000001011 \cdot 2^2 \quad B = +1,111011010 \cdot 2^{-2}$$

La diferencia de exponentes: $E_A - E_B = 2 - (-2) = 4$ nos dice que hay que desplazar la mantisa de B cuatro lugares a la derecha. Si partimos de:

$$B = +1,111011010 \cdot 2^{-2}$$

Al desplazar cuatro lugares (teniendo en cuenta el bit retenedor queda:

$$B = +0,000111101 \ 10 \ 1 \cdot 2^2$$

Se realiza la suma de mantisas (resta):

M_A	+ 1,000001011 00 0	
$+M_B$	- 0,000111101 10 1	
$A + B$	<u>+ 0,111001101 01 1</u>	$\cdot 2^2$
$A + B$	+ 1,110011010 11 0	$\cdot 2^1$
	+	
	<u>1</u>	
$A + B$	+ 1,110011011 01 0	$\cdot 2^1$

Normalizacion
Redondeo

$$A + B = +1,110011011 \cdot 2^1 = 0 \ 100001 \ 110011011 = \text{H}'439\text{B}$$

d) Determinación del error.

Hacemos la suma en decimal.

$$A + B = 4,0859375 + (-0,481445312) = 3,604491188$$

Pasamos a decimal los resultados de las sumas y calculamos los errores:

Formato 1.

$$A + B = 0000000011,100111 = 3,609375$$

$$\text{Error} = |3,604491188 - 3,609375| = 4,88 \cdot 10^{-3}$$

Formato 2.

$$A + B = +1,110011011 \cdot 2^1 = 11,10011011 = 3,60546875$$

$$\text{Error} = |3,604491188 - 3,60546875| = 9,78 \cdot 10^{-4}$$

1 (5 puntos) Programe en ensamblador del 88110 la función compacta que recibe cuatro parámetros en la pila:

- *v*: Es una matriz de enteros de *m* filas por *n* columnas almacenada por filas. Se pasa por dirección.
- *p*: Es una lista compacta que contiene elementos de una matriz. Se pasa por dirección. Cada uno de los elementos de la lista se compondrá de tres campos enteros:
 - *elemento*: Contiene el elemento que pertenece a la matriz.
 - *fila*: Indica la fila a la que pertenece el elemento. Las filas comienzan a numerarse en la posición 0.
 - *columna*: Indica la columna a la que pertenece el elemento. Las columnas comienzan a numerarse en la posición 0.

El final de la lista está indicado con el el valor -1 contenido en el campo fila.

- *m*: Es un entero que contiene el número de filas de la matriz *v*. Se pasa por valor.
- *n*: Es un entero que contiene el número de columnas de la matriz *v*. Se pasa por valor.

Esta función rellenará la lista apuntada por *p* con los elementos no nulos de la matriz *v*. Suponga que los valores *n* y *m* son mayores que 0.

El resultado del ejemplo que se muestra en la figura proviene de una matriz en la que su elemento (0, 5) contiene el valor 15, el (1, 5) el valor 20 y el (4, 8) el valor 40. El resto de los elementos de la matriz son 0.

P	20	1	5	15	0	5	40	4	8	5	-1	-1
---	----	---	---	----	---	---	----	---	---	---	----	----

SOLUCIÓN

El programa que se muestra a continuación recorre la matriz mediante dos bucles anidados. Comprueba si el elemento (*r6*) es no nulo y si es así almacena el elemento, la fila (*r4*) y la columna (*r5*) en el vector de salida *p* (*r21*).

```

PUSH:  MACRO (ra)
        subu  r30,r30,4
        st    ra,r30,0
ENDMACRO
POP:    MACRO (ra)
        ld    ra,r30,0
        addu  r30,r30,4
ENDMACRO

compacta: PUSH(r1)
        ld r20, r30, 4      ; r20: puntero a v
        ld r21, r30, 8      ; r21: puntero a p
        ld r2, r30, 12      ; r2: filas
        ld r3, r30, 16      ; r3: columnas
        or r4, r0, r0
        or r5, r0, r0
bucle:  ld r6, r20, r0
        cmp r7, r6, r0      ; Si el elemento es 0
        bbl eq, r7, cont
        st r6, r21, r0      ; Almacena elemento,
        st r4, r21, 4        ; fila y columna en p
        st r5, r21, 8
        addu r21, r21, 12    ; Incrementa el puntero a p
cont:   addu r5, r5, 1        ; Incrementa columnas
        addu r20, r20, 4    ; Incrementa el puntero a v
        cmp r7, r5, r3      ; Si es final de fila

```

```
bb1 ne, r7, bucle
addu r4, r4, 1           ; Incrementa filas
or r5, r0, r0
cmp r7, r4, r2           ; Si es final de matriz
bb1 ne, r7, bucle
or.c r7, r0, r0
st r7, r21, 4            ; -1 en los campos
st r7, r21, 8            ; fila y columna
POP(r1)
jmp(r1)
```

2 (5 puntos)

a) *Expresa a nivel RT (transferencia entre registros) las operaciones elementales que se producen en cada uno de los ciclos de la ejecución de la instrucción de una palabra BR \$desplazamiento. Incluya la fase de fetch.*

Para la estructura del computador especificada en el enunciado, las operaciones elementales durante la ejecución de la instrucción, son las siguientes:

```
c1: AR<- PC ; fase de fetch
c2: PC<- PC+1; ICM, R/W
c3: DR<- M(AR)
c4: RI<- DR
c5: Decodificación
c6: PC<- PC+RI(desplazamiento); fase de ejecución
```

a) *Represente en la figura 2 el cronograma de la ejecución de la instrucción anterior. Debe incluir todas las señales de control que se activan en cada ciclo de reloj, incluyendo la fase de fetch. Considere que la decodificación dura un ciclo completo.*

En la figura correspondiente a la plantilla del cronograma, se muestran a continuación las señales de control activadas durante los seis ciclos de ejecución de la instrucción:

ESTRUCTURA DE COMPUTADORES
SEGUNDO PARCIAL (14 de diciembre de 2010)

--	--	--

Apellidos, Nombre..... N° de Matrícula.....

1 Se tiene un PC con una unidad de disco duro con las siguientes características:

- 6,25 GBytes.
- 16 superficies con 2.048 pistas cada una (0-2.047).
- Cada pista tiene 200 sectores (0-199).
- Cada sector consta de 1.024 bytes de información neta y 1.280 de información bruta.
- Velocidad de rotación 6.000 rpm.
- En el movimiento de la cabeza de una pista a otra se tarda $0,025 \cdot n + 2$ ms, donde n es el número de pistas que separan la pista origen y la destino.

a) (1 punto) Calcule el tiempo de acceso medio de esta unidad de disco.

$$t_{\text{macc}} = t_{\text{mpos}} + t_{\text{est}} + t_{\text{mlat}} = \frac{2.047 \times 0,025}{2} + 2 + \frac{60}{6.000 \times 2} = 25,5875 + 2 + 5 = 32,5875 \text{ ms}$$

b) (1 punto) Determine el número de sector absoluto (lógico) correspondiente a las coordenadas geométricas (CHS) (1.234, 5, 67) y las coordenadas geométricas del sector absoluto n° 641.853.

$$(1.234, 5, 67) \Rightarrow 1.234 \times 200 \times 16 + 5 \times 200 + 67 = 3.949.867$$

$$641.853 \div (200 \times 16) = 200 \quad 641.853 \text{ módulo } (200 \times 16) = 1.853$$

$$1.853 \div 200 = 9 \quad 1.853 \text{ módulo } 200 = 53$$

$$641.853 \Rightarrow (200, 9, 53)$$

c) (1 punto) Suponiendo que en el instante $t=0$ las cabezas de la unidad se encuentran sobre el sector 100 del cilindro 1.800, calcule en qué instante termina una operación de lectura del sector (1.200, 8, 60) que comienza en el instante $t=13$ ms.

El tiempo que tarda el disco en dar una vuelta es $t_{\text{vuelta}} = \frac{60}{3.000} \text{ s} = 10 \text{ ms}$. El tiempo que emplea en recorrer un sector es $10/200=0,05 \text{ ms}$.

La operación de E/S comienza en el instante $t=13 \text{ ms}$. Su tiempo de búsqueda es $(1.800 - 1.200) \times 0,025 + 2 = 17 \text{ ms}$. Durante los $13 + 17 = 30 \text{ ms}$ el disco da 3 vueltas completas dejando las cabezas situadas nuevamente sobre el sector 100. El tiempo de latencia correspondiente al sector 60 es $(200 - 100 + 60) \times 0,05 = 8 \text{ ms}$. Por tanto, la operación de lectura termina en el instante $t = 13 + 17 + 8 + 0,05 = 38,05 \text{ ms}$.

2 Sea un computador con una CPU capaz de ejecutar 1.000 MIPS y con un sistema de interrupciones de varios niveles de prioridad que permite anidamiento e inhibición selectiva y cuya secuencia de reconocimiento de interrupciones (SRI) tiene una duración equivalente a 5 instrucciones. Este computador, con ancho de palabra de 32 bits, tiene conectado una pantalla gráfica.

Las características de la pantalla gráfica y de su módulo de E/S son las siguientes:

- Resolución 625×400 píxeles, 2^{32} colores por píxel y frecuencia de refresco vertical 100 Hz. Es decir, su velocidad de transferencia es de 10^8 bytes/s y su tamaño de bloque 10^6 bytes.
- Registro de datos de 32 bits.

a) (1 punto) Calcule el número máximo de instrucciones que puede tener la rutina de tratamiento de interrupciones (RTI) de este dispositivo.

$$\text{Frecuencia de interrupciones} = \frac{10^8 \text{ B/s}}{4 \text{ B/interr}} = 25 \cdot 10^6 \text{ interr/s}$$

$$25 \cdot 10^6 \text{ interr/s} \times (5 + X) \text{ instr/interr} \leq 10^9 \text{ instr/s}$$

$$125 + 25 \cdot X \leq 1000$$

$$X \leq 35$$

Este computador dispone de un disco duro con las siguientes características:

- Velocidad de transferencia: $16 \cdot 10^6$ bytes/s.
- Tiempo de acceso: 4 ms.
- Buffer de 4 registros de datos de 32 bits.
- Tamaño de bloque: 1024 bytes.

b) (1 punto) Si ambos dispositivos están conectados por interrupciones, determine cuál de los dos deberá tener más prioridad.

$$\text{Frecuencia de interrupciones HD} = \frac{15 \cdot 10^6 \text{ B/s}}{4 \cdot 4 \text{ B/interr}} = 10^6 \text{ interr/s}$$

Por lo tanto debe tener mayor prioridad la pantalla gráfica.

Esta forma de operación tiene las siguientes características:

- Las rutinas de inicio de operación para ambos dispositivos constan de 200 instrucciones.
- La rutina de tratamiento de las interrupciones de la pantalla consta de 25 instrucciones.
- La rutina de tratamiento de las interrupciones del disco duro consta de 45 instrucciones.
- Las rutinas de fin de operación para ambos dispositivos constan de 100 instrucciones.

c) (2 puntos) Calcule el tiempo de CPU que se consume para realizar una operación en cada dispositivo.

$$\text{Tiempo de CPU Pantalla} = 200 \text{ instr} + \frac{10^6 \text{ B}}{4 \text{ B/interr}} \times (5 + 25) \text{ instr/interr} + 100 \text{ instr} = 7,5 \cdot 10^6 \text{ instr} \rightarrow 7,5 \text{ ms}$$

$$\text{Tiempo de CPU HD} = 200 \text{ instr} + \frac{1024 \text{ B}}{4 \cdot 4 \text{ B/interr}} \times (5 + 45) \text{ instr/interr} + 100 \text{ instr} = 3.500 \text{ instr} \rightarrow 3,5 \mu\text{s}$$

d) (1 punto) Si ambos dispositivos operan simultáneamente, calcule la capacidad de procesamiento que necesitan.

Se calcula a partir de sus frecuencias de interrupción:

$$10^6 \text{ interr/s} \times (5 + 45) \text{ instr/interr} + 25 \cdot 10^6 \text{ interr/s} \times (5 + 25) \text{ instr/interr} = 50 \text{ MIPS} + 750 \text{ MIPS} = 800 \text{ MIPS}$$

Ya que esta capacidad de procesamiento es cercana a la de la CPU, se conecta la pantalla gráfica para operar mediante acceso directo a memoria (DMA).

Las características de esta forma de operación son las siguientes:

- Las rutinas de inicio y fin de la operación para la transferencia de una imagen a la pantalla tienen el mismo número de instrucciones.
- El protocolo de concesión y liberación de los buses dura 2 ns.
- El ciclo de acceso a memoria tiene una duración de 3 ns.

e) (2 puntos) Calcule el porcentaje de tiempo de CPU que se consume durante una operación de la pantalla mediante DMA.

El tiempo total de robo de ciclo es: $2 \text{ ns} + 3 \text{ ns} = 5 \text{ ns} \rightarrow 5 \text{ instr}$

$$\begin{aligned} \text{Tiempo de CPU Pantalla} &= 200 \text{ instr} + \frac{10^6 \text{ B}}{4 \text{ B/robo}} \times 5 \text{ instr/robo} + (5 + 100) \text{ instr} = 1.250.305 \text{ instr} \\ &\rightarrow 1.250.305 \text{ ns} \end{aligned}$$

$$\begin{aligned} \text{Tiempo operación Pantalla} &= 200 \text{ instr} + \frac{10^6 \text{ B}}{10^8 \text{ B/s}} \times 10^9 \text{ instruc/s} + 5 \text{ instr} + (5 + 100) \text{ instr} = 10.000.310 \text{ instr} \\ &\rightarrow 10.000.310 \text{ ns} \end{aligned}$$

$$\% \text{Uso de CPU} = \frac{1.250.305}{10.000.310} \times 100 = 12,5 \%$$

NOTAS: 10 de Enero de 2011
REVISIÓN: 12 de Enero de 2011

DURACIÓN: 75 minutos
PUNTUACIÓN: Especificada en cada apartado

ESTRUCTURA DE COMPUTADORES
SEGUNDO PARCIAL (27 de mayo de 2011)

--	--

--

Apellidos, Nombre..... N° de Matrícula.....
Responda en esta misma hoja, utilizando únicamente el espacio asignado para cada pregunta.

1 (3 puntos) Se tiene un PC con una unidad de disco duro con las siguientes características:

- 15,625 GBytes de información neta ($15,625 \cdot 2^{30}$ bytes).
- 16 superficies con 2.048 pistas cada una (0-2.047).
- Cada sector consta de 1.024 bytes de información neta y 1.280 de información bruta.
- Velocidad de rotación 6.000 rpm.
- En el movimiento de las cabezas de una pista a otra consecutiva se tarda 0,02 ms.
- El tiempo de estabilización de las cabezas es de 3 ms.

a) Calcule la **velocidad de transferencia** de esta unidad de disco.

$$N_{\text{sec}} = 15,625 \text{ Gbytes} / (16 \times 2.048 \times 1.024) = 500$$

$$V_{\text{rotacion}} = 6.000/60 = 100 \text{ r.p.s}$$

$$V_{\text{transf}} = 500 \times 1.280 \times 100 = 64 \cdot 10^6 \text{ bytes/s}$$

En el tiempo $t=0$ las cabezas de grabación se encuentran sobre el sector 120 del cilindro 1.000. En dicho instante da comienzo la operación de lectura del sector absoluto 8.005.370 y, a continuación, se ordena la escritura del sector 4.007.871.

b) Calcule el **tiempo de acceso** de la operación de escritura del sector 4.007.871.

$$8.005.370 / (16 \cdot 500) = 1.000$$

$$8.005.370 \bmod (16 \cdot 500) = 5.370$$

$$5.370 / 500 = 10$$

$$5.370 \bmod 500 = 370$$

$$\text{CHS}(8.005.370) \rightarrow (1.000, 10, 370)$$

$$4.007.871 / (16 \cdot 500) = 500$$

$$4.007.871 \bmod (16 \cdot 500) = 7.871$$

$$7.871 / 500 = 15$$

$$7.871 \bmod 500 = 371$$

$$\text{CHS}(4.007.871) \rightarrow (500, 15, 371)$$

$$T_{\text{vuelta}} = 1 / (100 \text{ r.p.s}) = 10 \text{ ms}$$

$$T_{\text{sector}} = 10 / 500 = 0,02 \text{ ms}$$

$$T_{\text{busqueda_sector1}} = 0 \text{ ms}$$

$$T_{\text{latencia_sector1}} = (370 - 120) \times 0,02 = 5 \text{ ms}$$

Tras leer el primer sector, las cabezas se encuentran ante el sector 371 del cilindro 1.000.

$$T_{\text{busqueda_sector2}} = (1.000 - 500) \times 0,02 + 3 = 13 \text{ ms}$$

$$\text{Sector_actual} \rightarrow (13/0,02 + 371) \bmod 500 = 21$$

$$T_{\text{latencia_sector2}} = (371 - 21) \times 0,02 = 7 \text{ ms}$$

$$T_{\text{acceso_sector2}} = 13 + 7 = 20 \text{ ms}$$

2 (7 puntos) Sea un computador con una CPU capaz de ejecutar 1.000 MIPS y con un sistema de interrupciones de varios niveles de prioridad que permite anidamiento e inhibición selectiva y cuya secuencia de reconocimiento de interrupciones (SRI) tiene una duración equivalente a 5 instrucciones. Este computador, con ancho de palabra de 32 bits, tiene conectado una unidad de disco duro con las siguientes características:

- Velocidad de transferencia: $16 \cdot 10^6$ bytes/s.
- Tiempo medio de acceso: 4 ms.
- Buffer de 4 registros de datos de 32 bits.
- Tamaño del sector: 1.024 bytes.

El módulo de E/S de la unidad de disco opera mediante interrupciones con los siguientes parámetros:

- Las rutinas de inicio y fin de una operación de E/S para la lectura o escritura de un sector constan de 125 y de 75 instrucciones respectivamente.
- Su rutina de tratamiento de las interrupciones consta de 45 instrucciones.

a) Si la unidad de disco opera por interrupciones, determine cuál es el máximo tiempo de respuesta por parte de la CPU en atender sus solicitudes de interrupción (tiempo de respuesta \equiv tiempo que puede tardar en responder).

$$T_{\text{instruccion}} = 1/1.000 \cdot 10^6 = 1 \text{ ns} \quad T_{\text{SRI}} = 5 \times 1 \text{ ns} = 5 \text{ ns} \quad T_{\text{RSI}} = 45 \times 1 \text{ ns} = 45 \text{ ns}$$

$$T_{\text{entre_interrupciones}} = (4 \times 4) \text{ bytes} / 16 \cdot 10^6 \text{ bytes/s} = 1.000 \text{ ns}$$

$$T_{\text{respuesta}} = 1.000 - (5 + 45) = 950 \text{ ns}$$

b) Calcule cuánto tiempo dura una operación de E/S para la lectura de un sector del disco.

$$t_{\text{inst}} = \frac{1}{10^9} \text{ s} = 1 \text{ ns}$$

$$\begin{aligned} t_{\text{op}} &= t_{\text{ini}} + t_{\text{acc}} + t_{\text{trans}} + t_{\text{int}} + t_{\text{fin}} = \\ &= 125t_{\text{inst}} + 4 \text{ ms} + \frac{1.024}{16 \cdot 10^6} \text{ s} + (5 + 45)t_{\text{inst}} + 75t_{\text{inst}} = \\ &= 125 \text{ ns} + 4 \text{ ms} + 64 \mu\text{s} + 50 \text{ ns} + 75 \text{ ns} = 4.064.250 \text{ ns} \end{aligned}$$

Este computador dispone también de una conexión a una red Ethernet con las siguientes características:

- Operación por ráfagas de acceso directo a memoria (DMA).
- Velocidad de transmisión de 1 Gigabit por segundo (10^9 bits/s).
- Buffer de 16 registros de datos de 32 bits.
- Las rutinas de inicio y fin de una operación de E/S para la transferencia de un bloque de datos constan de 200 y de 100 instrucciones respectivamente.
- La rutina de tratamiento de las interrupciones de la red Ethernet consta de 75 instrucciones.
- El protocolo de concesión y liberación de los buses dura 4 ns.
- El ciclo de acceso a memoria tiene una duración de 1 ns.

c) Determine cuál de los dos periféricos deberá tener más prioridad para la atención de sus interrupciones.

Las interrupciones de la red ethernet indican que se ha completado una operación de E/S, mientras que las de la unidad de disco indican la llegada de nuevos datos que deben ser leídos antes de que se reciban los siguientes, en el caso de las operaciones de entrada, o la disponibilidad de los registros de datos para recibir los nuevos datos que deben ser transmitidos manteniendo la velocidad de transferencia, en el caso de las operaciones de salida.

Por tanto, dado que una atención insuficientemente rápida de las interrupciones no compromete la correcta finalización de las operaciones de E/S de la red ethernet y sí compromete las de la unidad de disco, estas últimas deben ser más prioritarias.

d) Calcule qué porcentaje de tiempo queda libre para la CPU durante una operación de E/S correspondiente a la recepción de un bloque de datos de 1.024 bytes.

$$\begin{aligned} t_{\text{op_eth}} &= t_{\text{ini}} + t_{\text{acc}} + t_{\text{trans}} + t_{\text{dma}} + t_{\text{int}} + t_{\text{fin}} = \\ &= 200t_{\text{inst}} + 0 + \frac{1.024 \times 8}{10^9} \text{ s} + (4 + 16 \times 1) \text{ ns} + (5 + 75)t_{\text{inst}} + 100t_{\text{inst}} = \\ &= 200 \text{ ns} + 0 + 8.192 \text{ ns} + 20 \text{ ns} + 80 \text{ ns} + 100 \text{ ns} = 8.592 \text{ ns} \end{aligned}$$

$$\begin{aligned} t_{\text{cpu_eth}} &= t_{\text{ini}} + N_{\text{dma}} \times t_{\text{dma}} + t_{\text{int}} + t_{\text{fin}} = \\ &= 200 \text{ ns} + \frac{1.024}{16 \times 4} \times 20 \text{ ns} + 80 \text{ ns} + 100 \text{ ns} = 700 \text{ ns} \end{aligned}$$

$$\%_{\text{cpu_libre}} = \frac{8.592 - 700 \text{ ns}}{8.592 \text{ ns}} \times 100 = 91,85 \%$$

e) Si ambos dispositivos operan simultáneamente, determine cuál es el máximo tiempo de respuesta por parte de la CPU en atender sus solicitudes de interrupción.

En el caso de la red ethernet, dado que las interrupciones señalan el final de las operaciones de E/S, el máximo tiempo de respuesta no está definido. En el caso de la unidad de disco, al tiempo calculado en el apartado a) hay que restar el tiempo que los ciclos de dma de la red ethernet retrasan la ejecución de la rutina de servicio de interrupción incluyendo la secuencia de reconocimiento de interrupción:

$$\%_{\text{cpu_libre_dma}} = \frac{t_{\text{trans}} - N_{\text{dma}} \times t_{\text{dma}}}{t_{\text{trans}}} \times 100 = \frac{8.192 - 16 \times 20}{8.192} = 96,09375 \%$$

$$t_{\text{int_dma}} = \frac{5+45}{0,9609375} = 52,03 \text{ ns} \quad t_{\text{resp}} = t_{\text{entre-interrupciones}} - t_{\text{int_dma}} = 1.000 - 52,03 = 947,97 \text{ ns}$$

f) Si la unidad de disco y la red Ethernet están operando continua y simultáneamente, calcule cuánto tiempo tarda en ejecutarse un programa que emplea 10 millones de instrucciones. Suponga que la red Ethernet siempre recibe bloques de 1.024 bytes.

$\%_{\text{cpu_udisco}} :$

$$t_{\text{op_udisco}} = 4.064.250 \text{ ns (apartado b))}$$

$$t_{\text{cpu_udisco}} = t_{\text{ini}} + N_{\text{int}} \times t_{\text{int}} + t_{\text{fin}} = 125 \text{ ns} + \frac{1.024}{4 \times 4} \times (5 + 45) \text{ ns} + 75 \text{ ns} = 3.400 \text{ ns}$$

$$\%_{\text{cpu_udisco}} = \frac{3.400}{4.064.250} \times 100 = 0,08366 \%$$

$$\%_{\text{cpu_ethernet}} = \frac{700}{8.592} \times 100 = 8,1471 \% \text{ (apartado d))}$$

$$\%_{\text{cpu_libre}} = 100 - 0,08366 - 8,1471 = 91,7692 \%$$

$$t_{\text{programa}} = \frac{10 \cdot 10^6 \times 1 \text{ ns}}{0,917692} = 10.896.902,23 \text{ ns}$$

NOTAS: 7 de junio de 2011
REVISIÓN: 9 de junio de 2011

DURACIÓN: 75 minutos
PUNTUACIÓN: Especificada en cada apartado

1 Sea una unidad de disco duro con las siguientes características:

- 8 superficies con 20.000 pistas cada una.
- Cada pista tiene 100 sectores.
- Cada sector consta de 1.024 bytes de información neta y 1.250 de información bruta.
- Velocidad de rotación: 12.000 rpm.
- El tiempo de mover la cabeza de una pista a otra consecutiva es $1\mu s$
- El tiempo de estabilización de las cabezas es de 2 ms.

a) (1 punto) Calcule la velocidad de transferencia y el tiempo medio de acceso a un sector.

b) (1 punto) Determine el número de sector absoluto (lógico) correspondiente a las coordenadas geométricas (CHS) (1.234, 5, 67) y las coordenadas geométricas del sector absoluto nº 654.321.

c) (1 punto) Suponiendo que en el instante $t=0$ las cabezas de la unidad se encuentran al comienzo del sector 20 del cilindro 10.000, calcule en qué instante termina la operación de lectura del sector (16.000, 4, 60).

SOLUCIÓN

a)

$$v_{transf} = \frac{12.000 \text{ pistas/minuto}}{60 \text{ s/minuto}} \times 100 \text{ sectores/pista} \cdot 1.250 \text{ bytes/sector} = 25 \cdot 10^6 \text{ bytes/s}$$

$$t_{lat} = \frac{60 \text{ s/minuto}}{12.000 \text{ revoluciones/minuto}} = 5 \text{ ms/revolución} \rightarrow \bar{t}_{lat} = 2,5 \text{ ms}$$

$$\bar{t}_{búsq} = \bar{t}_{pos} + t_{est} = 0,001 \cdot \bar{n} \text{ ms} + 2 \text{ ms} = 0,001 \cdot 10.000 \text{ ms} + 2 \text{ ms} = 12 \text{ ms}$$

$$\bar{t}_{acc} = \bar{t}_{búsq} + \bar{t}_{lat} = 12 \text{ ms} + 2,5 \text{ ms} = 14,5 \text{ ms}$$

b)

$$(1.234, 5, 67) \rightarrow 1.234 \times 100 \times 8 + 5 \times 100 + 67 = 987.767$$

$$\begin{array}{rcl} \frac{654.321}{100 \cdot 8} & = & 817 \\ 654.321 \bmod 100 \cdot 8 & = & 721 \\ \frac{721}{100} & = & 7 \\ 721 \bmod 100 & = & 21 \\ 641.853 & \rightarrow & (817, 7, 21) \end{array}$$

c)

En $t = 0,001 \times 6.000 \text{ ms} + 2 \text{ ms} = 8 \text{ ms}$ se encuentra en el cilindro 16.000 y como gira a razón de una revolución cada 5 ms, habrá girado 1,6 vueltas, es decir 160 sectores. De este modo las cabezas están al comienzo del sector $20 + 160 \bmod 100 = 80$.

Para alcanzar el sector 60 habrán de pasar $60 - 80 + 100 = 120$ sectores. Es decir 1,2 vueltas o 1,2 vueltas $\times 5 \text{ ms/vuelta} = 6 \text{ ms}$. Luego en 14 ms se encuentra al comienzo del sector 60.

$$t_{\text{sector}} = \frac{5 \text{ ms/pista}}{100 \text{ sectores/pista}} = 50 \mu\text{s}$$

$$t_{\text{acc}} = 8 \text{ ms} + 6 \text{ ms} + 0,05 \text{ ms} = 14,05 \text{ ms}$$

2 Sea un computador con una CPU de 32 bits con un procesador capaz de ejecutar 2.000 MIPS y cuya secuencia de reconocimiento de interrupciones (SRI) tiene una duración equivalente a 4 instrucciones. Este computador de 32 bits dispone de varias unidades de discos duros con las siguientes características:

- Velocidad de transferencia: $25 \cdot 10^6$ bytes/s.
- Tiempo medio de acceso: 14,5 ms.
- Tamaño de bloque: 1.024 bytes.

Los módulos de E/S de las unidades de disco operan mediante interrupciones con los siguientes parámetros:

- Registro de datos de 32 bits.
- La rutina de inicio de operación consta de 150 instrucciones.
- La rutina de tratamiento de las interrupciones del disco duro consta de 96 instrucciones.
- La rutina de fin de operación consta de 200 instrucciones.

a) (1 punto) Calcule cuántas unidades de discos pueden operar en paralelo.

b) (2 puntos) Indique qué modificaciones se deben hacer a los módulos de E/S para reducir la sobrecarga debida al tratamiento de las interrupciones y, así, permitir que pueda operar simultáneamente el doble del número de discos calculado en el apartado anterior. Calcule cuántas instrucciones podría tener como máximo las nuevas rutinas de tratamiento de las interrupciones.

c) (2 puntos) Calcule el tiempo de CPU que consume una operación de lectura de un bloque de un disco, suponiendo que con el buffer de 2 registros de 32 bits las rutinas de tratamiento de interrupciones constan de 100 instrucciones.

Para reducir la capacidad de procesamiento necesaria para las operaciones de E/S, los discos se conectan para operar mediante acceso directo a memoria (DMA). Suponga que:

- Los módulos de E/S disponen de un buffer de 4 registros de datos de 32 bits.
- Las rutinas de inicio y fin de una operación de E/S para la transferencia de un bloque de datos constan de 100 y de 200 instrucciones respectivamente.
- El protocolo de concesión y liberación de los buses dura 1 ns cada uno.
- El ciclo de acceso a memoria tiene una duración de 2 ns.

d) (2 puntos) Calcule el porcentaje de tiempo de CPU que se consume durante una operación de lectura de un bloque de un disco.

SOLUCIÓN

a) La capacidad de proceso que requiere el tratamiento de las interrupciones de una unidad de disco se calcula a partir de su frecuencia de petición:

$$\text{Frecuencia de petición} = \frac{v_{\text{transf}}}{\text{Bytes por petición}} = \frac{25 \cdot 10^6 \text{ bytes/s}}{4 \text{ bytes/petición}} = 6,25 \cdot 10^6 \text{ peticiones/s}$$

Cada tratamiento de interrupción supone realizar la SRI y ejecutar la RTI, lo que equivale a 100 instrucciones:

$$\text{Instrucciones por segundo} = 6,25 \cdot 10^6 \text{ peticiones/s} \times 100 \text{ instrucciones/petición} = 625 \cdot 10^6 \text{ instrucciones/s}$$

Cada disco necesita una capacidad de cómputo de 625 MIPS, por lo tanto solo podrán operar en paralelo 3 unidades con este procesador de 2.000 MIPS.

b)

Dotando a los módulos de E/S de un buffer de 2 registro de datos de 32 bits se reduciría la frecuencia de petición de interrupciones a la mitad.

$$\text{Frecuencia de petición} = \frac{25 \cdot 10^6 \text{ bytes/s}}{8 \text{ bytes/petición}} = 3,125 \cdot 10^6 \text{ peticiones/s}$$

Para que puedan operar simultáneamente, la capacidad de procesamiento de las 6 unidades de disco debe ser menor que la del procesador:

$$\begin{aligned} \text{Instrucciones por segundo} &= \\ 6 \times 3,125 \cdot 10^6 \text{ peticiones/s} \times (4 + X) \text{ instrucciones/petición} &\leq 2.000 \cdot 10^6 \text{ instrucciones/s} \\ 18,750 \cdot 10^6 (4 + X) \text{ instrucciones/s} &\leq 2.000 \cdot 10^6 \text{ instrucciones/s} \\ 18,750 \cdot (4 + X) &\leq 2.000 \\ X &\leq 102.7 \rightarrow 102 \text{ instrucciones} \end{aligned}$$

c)

$$\begin{aligned} t_{\text{cpu_HD_interrupciones}} &= t_{\text{ini}} + N_{\text{interrupciones}} \times t_{\text{interrupción}} + t_{\text{fin}} = \\ &= \frac{150 \text{ inst}}{2 \text{ inst/ns}} + \frac{1024 \text{ bytes}}{2 \cdot 4 \text{ bytes}} \times \frac{(4 + 100) \text{ inst}}{2 \text{ inst/ns}} + \frac{4 + 200 \text{ inst}}{2 \text{ inst/ns}} = \\ &= 6.831 \text{ ns} \end{aligned}$$

d)

$$\begin{aligned} t_{\text{op_HD_ráfaga}} &= t_{\text{ini}} + t_{\text{acc}} + t_{\text{trans}} + t_{\text{dma}} + t_{\text{int}} = \\ &= \frac{100 \text{ inst}}{2 \text{ inst/ns}} + 14,5 \cdot 10^6 \text{ ns} + 50.000 \text{ ns} + (1 \text{ ns} + 4 \cdot 2 \text{ ns} + 1 \text{ ns}) + \frac{(4 + 200) \text{ inst}}{2 \text{ inst/ns}} = \\ &= 14.550.162 \text{ ns} \\ t_{\text{cpu_HD_ráfaga}} &= t_{\text{ini}} + N_{\text{dma}} \times t_{\text{dma}} + t_{\text{int}} = \\ &= \frac{100 \text{ inst}}{2 \text{ inst/ns}} + \frac{1024 \text{ bytes}}{4 \cdot 4 \text{ bytes}} \times (1 \text{ ns} + 4 \cdot 2 \text{ ns} + 1 \text{ ns}) + \frac{(4 + 200) \text{ inst}}{2 \text{ inst/ns}} = \\ &= 792 \text{ ns} \\ \%_{\text{cpu_ráfaga}} &= \frac{792 \text{ ns}}{14.550.162 \text{ ns}} \times 100 = \\ &= 0,005 \% \end{aligned}$$

1 Un computador cuenta con los dos formatos de representación siguientes :

- Formato 1. Coma fija: 16 bits en complemento a dos con la coma entre los bits 3 y 4, por ejemplo 010001101110,1001.
- Formato 2. Coma flotante: 16 bits, el bit superior es el signo, los siete siguientes el exponente, representado en exceso a 64 y los 8 siguientes la magnitud de la mantisa, que está representada en signo magnitud, con bit implícito y la coma a la izquierda de este.

a) Determine el rango y la resolución de ambos formatos.

b) Se tienen dos datos A y B, A corresponde a la cadena de 16 bits H'450D en su representación en coma flotante (formato 2) y B = -1,46 (valor decimal). Determine el valor decimal de A y represente A y B en ambos formatos.

c) Realice paso a paso la operación A+B en ambos formatos, indicando el valor decimal del resultado. En el caso de coma flotante utilice dos bits de guarda y un bit retenedor, así como redondeo al más próximo.

SOLUCIÓN

a) Rango y resolución de los formatos.

Formato 1.

$$\begin{cases} 1000\ 0000\ 0000,0000 & \rightarrow -2^{11} \\ 0111\ 1111\ 1111,1111 & \rightarrow 2^{11} - 2^{-4} \end{cases}$$

$$\text{Rango} = [-2^{11}, 2^{11} - 2^{-4}]$$

$$\text{Resolución} = 2^{-4}$$

Formato 2.

Exponente: $[-64, 63]$. Como hay que reservar el exponente -64 para la representación del cero, el rango del exponente queda: $[-63, 63]$.

$$\text{Mantisa: } \pm \begin{cases} ,100000000 & \rightarrow 2^{-1} \\ ,111111111 & \rightarrow 1 - 2^{-9} \end{cases}$$

$$\text{Rango} = \pm [2^{-1} \cdot 2^{-63}, (1 - 2^{-9}) \cdot 2^{63}] \cup 0$$

$$\text{Resolución} = 2^{-9} \cdot 2^E$$

b) Valor decimal y representación en coma fija.

$$A = \text{H}'450\text{D} = 0\ 1000101\ 00001101 = +,100001101 \cdot 2^5 = 10000,1101 = 16,8125$$

$$B = -1,46 = -1,01110101 = -,101110101 \cdot 2^1$$

Nótese que $0,46 \times 16 = 7,36$ y $0,36 \times 16 = 5,76$, siendo suficientes los ocho bits de la parte fraccionaria de B para su representación en ambos formatos.

Representaciones en el formato de coma fija:

$$A = 000000010000,1101 = \text{H}'010\text{D}$$

$$B = -000000000001,0111 = 111111111110,1001 = \text{H}'\text{FFE}9$$

Representaciones en el formato de coma flotante:

$$A = 0\ 1000101\ 00001101 = \text{H}'450\text{D}$$

$$B = 1\ 1000001\ 01110101 = \text{H}'\text{C175}$$

c) Suma A + B.

Formato 1.

$$\begin{array}{r} A \\ +B \\ \hline A+B \end{array} \quad \begin{array}{r} 000000010000,1101 \\ 111111111110,1001 \\ \hline 000000001111,0110 \end{array}$$

$$A + B = 000000001111,0110 = \text{H}'00\text{F5} = 15,375$$

Formato 2.

Los números a sumar son:

$$A = +,100001101 \cdot 2^5 \quad B = -,101110101 \cdot 2^1$$

La diferencia de exponentes: $E_A - E_B = 5 - 1 = 4$ nos dice que hay que desplazar la mantisa de B cuatro lugares a la derecha. Si partimos de:

$$B = -,101110101 \cdot 2^1$$

Al desplazar cuatro lugares (teniendo en cuenta el bit retenedor) queda:

$$B = +,000010111 \ 01 \ 1 \cdot 2^5$$

Se realiza la suma de mantisas (resta):

M_A	$+,100001101 \ 00 \ 0$	
$+M_B$	$-,000010111 \ 01 \ 1$	
$A + B$	$+,011110101 \ 10 \ 1 \cdot 2^5$	
$A + B$	$+,111101011 \ 01 \ 0 \cdot 2^4$	<i>Normalizacion</i>
	$+$	<i>Redondeo</i>
$A + B$	$+,111101011 \ 11 \ 0 \cdot 2^4$	

$$A + B = +111101011 \cdot 2^4 = 15,31640625 = 0 \ 1000100 \ 11101011 = \text{H}'44\text{EB}$$

1 Indique, justificando su respuesta, si las siguientes afirmaciones son verdaderas o falsas:

- a) Los procesadores actuales son más inteligentes, lo que les permite conectarse directamente a gran variedad de periféricos, sin necesidad de módulos de E/S.
- b) El registro de instrucción es un registro de propósito específico que contiene la dirección de la siguiente instrucción a ejecutar.
- c) La memoria de un procesador contiene tanto datos como instrucciones, pero los etiqueta para diferenciarlos y que no se produzcan errores como intentar ejecutar un dato.
- d) El registro de datos de memoria es un registro transparente al usuario, ya que el usuario no tiene por qué conocer su existencia, en el que se almacena el dato que se quiere leer o escribir en memoria.

SOLUCIÓN

- a) Falso. Los módulos de E/S son necesarios, para unificar la visión hardware de las diferentes características de los periféricos.
- b) Falso. El registro de instrucción contiene la instrucción que se está ejecutando.
- c) Falso. La memoria contiene datos e instrucciones, pero no están etiquetados. Al intentar ejecutar un dato se producirá un error.
- d) Verdadero. El registro de datos de memoria es un registro transparente, no referenciado de forma directa por las instrucciones, donde se almacena el dato a leer o escribir en memoria.

2 Programe en ensamblador del 88110 los fragmentos de la subrutina *SUBROUTINA* que se detallan a continuación, teniendo en cuenta que el primer elemento de los vectores es el que su índice es el 0, los parámetros se pasan en la pila y las variables locales se almacenan en el marco de pila. *TRANS* es una subrutina que aplica una transformación al elemento que se pasa como parámetro por dirección en la pila.

```

SUBROUTINA (int v[n], int n)
{
    int aux[n];          /* Variable local: Vector de n enteros auxiliar */
    int i = n;           /* Variable local: Entero inicializado a n */

    while (i > 0)
    {
        aux[n-i] = v[i-1]; /* En aux se genera el vector "invertido" */
        TRANS(&aux[n-i]); /* Se aplica una transformación al elemento del
                           vector invertido que se pasa por dirección */
        i = i - 1;         /* es decir, el primer elemento de v pasa a ser */
                           /* el último de aux */
    }
}

```

SOLUCIÓN

La solución del ejercicio se basa en reservar en la pila tantas palabras como indique *n* más una que contiene el contador *i*. Esta última es la que está más próxima al puntero de marco de pila (*r31*), de tal forma que el comienzo del vector *aux* es la dirección contenida por el puntero de pila (*r30*). Una vez que se llama a *TRANS*, se recuperan *r3* con el valor de *i*, *r4* con *n* y *r20* con la dirección de comienzo de *v* puesto que *TRANS* es una subrutina que puede utilizar cualquier conjunto de registros y la salvaguarda de los mismos la debe realizar el programa llamante (*SUBROUTINA*).

```

SUBROUTINA: PUSH(r1)
            PUSH(r31)
            or r31,r30,r30
            ld r3, r31, 12      ; n

```



```

        add r4, r3, 1
        mulu r5, r4, 4
        subu r30,r30,r5      ; Se reservan n + 1 palabras como var. locales
        st r3, r31, -4      ; Se inicializa var. local i = n
bucle:  cmp r5,r3,r0
        bbl eq, r5, fin     ; Si i = 0 finaliza
        ld r4, r31, 12      ; Carga n
        sub r4, r4, r3      ; Calcula índice de aux
        sub r3, r3, 1
        st r3, r31, -4      ; i = i - 1
        mulu r3, r3, 4
        mulu r4, r4, 4
        ld r20, r31, 8      ; v
        ld r2, r20, r3      ; v[i-1]
        st r2, r30, r4      ; aux[n-i]
        addu r20,r30,r4
        PUSH(r20)
        bsr TRANS
        addu r30,r30,4
        ld r3, r31, -4      ; Recupera i
        br bucle
fin:

```

3 En la estructura de un computador con Unidad de Control *microprogramada* se presentan los siguientes retardos:

- operación elemental de mayor duración: 20 unidades de tiempo (ut)
- secuenciador: 5 ut
- memoria de control: 15 ut
- lectura o escritura de registros: 1 ut

a) Calcule razonadamente el ciclo de reloj del computador.

b) Si se sustituye la Unidad de Control por otra de tipo **cableada**. ¿Con qué ciclo de reloj debería trabajar ahora el computador? Justifique su respuesta.

SOLUCIÓN

a) Unidad de Control microprogramada:

El ciclo de reloj se calculará sumando los retardos de todos los elementos implicados en la ejecución de cada microinstrucción:

$$T_{ck} = t_{Reg.mdir} + t_{MC} + 2.t_{Reg.control} + t_{Op.maslarga} + t_{RE} + t_{SEC} + t_{Reg.mdir} = (1 + 15 + 2 + 20 + 1 + 5 + 1) = 45ut$$

b) Unidad de Control cableada:

En este caso, el ciclo de reloj corresponderá al retardo de la operación elemental de mayor duración:

$$T_{ck} = 20ut$$

4 Sea una unidad de disco duro de brazo móvil con las siguientes características:

- $40,96 \cdot 10^9$ bytes de capacidad bruta y 80 % de almacenamiento neto.
- 32 [0..31] superficies, 200 [0..199] sectores por pista y 512 bytes de información neta por sector.
- Velocidad de rotación: 12.000 rpm.
- El tiempo que emplea en mover la cabeza de un cilindro a otro consecutivo es 10 μ s.
- El tiempo de estabilización de las cabezas es 2 ms.

a) Calcule la velocidad de transferencia y el tiempo medio de acceso de la unidad de disco.

b) En el instante $t=0$ s la cabeza se encuentra situada al comienzo del sector absoluto 0. Si en $t=0$ s se ordena la lectura del sector (15, 450, 20), calcule el instante en el que finaliza la lectura de dicho sector.

Se conecta este disco a un computador de 1.000 MIPS mediante un módulo de entrada/salida con un registro de datos de 32 bits. Además, suponga que:

- La duración de la Secuencia de Reconocimiento de Interrupciones (SRI) es 5 ns.
- La rutina de programación de una operación de entrada/salida tiene 100 instrucciones.
- En todos los casos, la Rutina de Servicio de Interrupción tiene 50 instrucciones.
- El tiempo de un ciclo de acceso a memoria principal es de 2 ns.
- El protocolo de concesión/liberación de buses consume un total de 2 ns.

c) Calcule el número de discos como éste que pueden operar simultáneamente.

d) Calcule el tiempo total de CPU que se dedica a la lectura del sector si se opera mediante interrupciones.

e) Calcule el tiempo total de CPU que se dedica a la lectura del sector si se opera mediante acceso directo a memoria (DMA).

f) Calcule el tiempo de lectura del apartado b) suponiendo que se opera mediante DMA.

SOLUCIÓN

a) El cálculo de la velocidad de transferencia lo podemos hacer a partir de la capacidad bruta de una pista y la velocidad de rotación:

$$\begin{aligned} v_{\text{transf}} &= \text{pistas} - \text{por} - \text{segundo} \times \text{Capacidad bruta por pista} = \\ &= \frac{12.000 \text{ pistas/minuto}}{60 \text{ s/minuto}} \times 200 \text{ sectores/pista} \times \frac{512 \text{ bytes/sector}}{0,8} = 25,6 \cdot 10^6 \text{ bytes/s} \end{aligned}$$

Para calcular el tiempo medio de acceso necesitamos saber el número de cilindros que se pueden obtener a partir de la capacidad bruta:

$$\text{Capacidad bruta} = N_{\text{superficies}} \cdot N_{\text{cilindros}} \cdot \text{Capacidad bruta por pista} = 40,96 \cdot 10^9 \text{ Bytes}$$

$$N_{\text{cilindros}} = \frac{40,96 \cdot 10^9 \text{ Bytes}}{32 \text{ superficies} \times 128.000 \text{ Bytes/pista}} = 10.000 \text{ pistas/superficie}$$

$$t_{\text{lat}} = \frac{60 \text{ s/minuto}}{12.000 \text{ revoluciones/minuto}} = 5 \text{ ms/revolución} \rightarrow \bar{t}_{\text{lat}} = 2,5 \text{ ms}$$

$$\bar{t}_{\text{búsq}} = \bar{t}_{\text{pos}} + t_{\text{est}} = 0,001 \cdot \bar{n} \text{ ms} + 2 \text{ ms} = 0,01 \cdot 5.000 \text{ ms} + 2 \text{ ms} = 52 \text{ ms}$$

$$\bar{t}_{\text{acc}} = \bar{t}_{\text{búsq}} + \bar{t}_{\text{lat}} = 52 \text{ ms} + 2,5 \text{ ms} = 54,5 \text{ ms}$$

b) El brazo tendrá que moverse hasta el cilindro 450 y esperar a que pase el sector 20.

$$t_{\text{mover_450_cilindros}} = 450 \cdot 0,01 \text{ ms} + 2 \text{ ms} = 6,5 \text{ ms}$$

$$t_{\text{pasar_sector}} = \frac{60 \text{ s/minuto}}{12.000 \text{ pistas/minuto} \times 200 \text{ sectores/pista}} = 25 \mu\text{s}$$

Luego en $t = 6,5 \text{ ms}$ se encuentra en el cilindro 450 al comienzo del sector 6.500 $\mu\text{s}/25 \mu\text{s} = 260 \rightarrow 60$. Para leer el sector 20, el disco ha de girar 160 sectores.

$$t_{\text{pasar_160_sectores}} = 160 \text{ sectores} \times 25 \mu\text{s/sector} = 4 \text{ ms}$$

En $t = 10,5 \text{ ms}$ se encuentra en el cilindro 450 al comienzo del sector 20. Y en $t = 10,5 \text{ ms} + 0,025 \text{ ms} = 10,75 \text{ ms}$ se acaba la lectura del sector.

Como el procesador tiene una capacidad de 1.000 MIPS, cada instrucción se ejecuta en 1 ns. además, la SRI tiene una duración equivalente a la ejecución de 5 instrucciones.

c) Se calcula la capacidad de proceso que requiere la atención a las interrupciones de un disco y se obtiene el número de discos que pueden operar simultáneamente sin saturar al procesado .

$$\begin{aligned} \text{Capacidad_de_proceso} &= \text{Frecuencia_de_interrupciones} \times \text{instrucciones/interrupción} = \\ &= \frac{25,6 \cdot 10^6 \text{ bytes/s}}{4 \text{ bytes/interrupción}} \times (5 + 50) \text{ instrucciones/interrupción} = \\ &= 352 \cdot 10^6 \text{ instrucciones/s} \end{aligned}$$

$$\text{Número_de_discos} = \left\lceil \frac{10^9 \text{ instrucciones/s}}{352 \cdot 10^6 \text{ instrucciones/s}} \right\rceil = 2$$

d) En el caso de operación mediante interrupciones, el módulo de entrada/salida solicitará 128 interrupciones para transferir los 512 bytes del sector. Además hay que contabilizar la rutina de inicio de operación.

$$t_{\text{ocupCPU}} = 100 \text{ instrucciones} + 128 \text{ interrupciones} \times (5 + 50) \text{ instrucciones/interrupción}$$

e) En el caso de operación mediante robo de ciclo, el módulo de entrada/salida solicitará 128 veces los buses para transferir los 512 bytes del sector. Además hay que contabilizar la rutina de inicio de operación y la interrupción de finalización.

Cada vez que el módulo de entrada/salida solicita los buses la CPU se detiene durante 4 ns.

$$t_{\text{ocupCPU}} = \frac{100 \text{ instrucciones}}{10^9 \text{ instrucciones/s}} + 128 \text{ robos} \times 4 \cdot 10^{-9} \text{ s/robo} + \frac{(5 + 50) \text{ instrucciones}}{10^9 \text{ instrucciones/s}} = 667 \text{ ns}$$

f) En este caso el disco empezará a moverse tras la ejecución de la rutina de inicio, es decir en $t=100 \text{ ns}$.

De este modo 6,5 ms más tarde se encontrará en el cilindro 450 pero no al comienzo del sector 60 sino que habrá leído los primeros bytes de dicho sector. En cualquier caso, en $t=10 \text{ ms}$ se encontrará al comienzo de sector 0 y en $t=10,5 \text{ ms}$ se encontrará al comienzo del sector 20.

Por lo tanto, en 10,75 ms se acaba la lectura del sector y faltará el tiempo del último robo de ciclo y la rutina de tratamiento de interrupción.

$$t_{\text{operación}} = 10,75 \text{ ms} + 4 \cdot 10^{-9} \text{ s} + \frac{100 \text{ instrucciones}}{10^9 \text{ instrucciones/s}} = 10.750,104 \mu\text{s}$$

5 Un computador cuenta con un formato de coma flotante de 24 bits en el que el bit superior es el signo, los ocho siguientes el exponente, representado en exceso a 128 y los 15 siguientes la magnitud de la mantisa, que está representada en signo-magnitud, con bit implícito y la coma a la izquierda de éste.

a) Represente en el formato los números decimales: $A = -30,84$ y $B = 0,3244$.

b) Realice paso a paso la operación $A+B$ indicando el valor decimal del resultado. Utilice dos bits de guarda y un bit retenedor, así como redondeo al más próximo.

SOLUCIÓN

1.- Representación de los números.

$$A = -30,84 = -11110,110101110000 = -,1111011010111000 \cdot 2^5$$

$$A = 1\ 10000101\ 111011010111000 = \text{H}'\text{C2F6B8}$$

$$B = +0,3244 = +,01010011000010111 = +,1010011000010111 \cdot 2^{-1}$$

$$B = ,0\ 01111111\ 01001000010111 = \text{H}'\text{3FA617}$$

2.- Suma $A+B$.

Los números a sumar son: $A = -,1111011010111000 \cdot 2^5$ y $B = +,1010011000010111 \cdot 2^{-1}$

Se restan los exponentes: $E_A - E_B = 5 - (-1) = 6$.

Hay que desplazar la mantisa de B 6 lugares a la derecha. Usando dos bits de guarda y un retenedor queda:

$$B = +,0000001010011000\ 01\ 1 \cdot 2^{-5}$$

Hay que restar el valor absoluto de la mantisa de A menos el valor absoluto de la mantisa de B y poner el signo de la mantisa de A.

P	G	
,1111011010111000	00 0	
- ,0000001010011000	01 1	
,1111010000011111	01 1	
	1	
,1111010000011111		Esta normalizado.

$$A + B = -,1111010000100000 \cdot 2^5 = -,11110,10000100 = -30,515625.$$

En el formato:

$$A + B = 1\ 10000101\ 111010000100000 = \text{H}'\text{C2F420}$$

1 Indique, justificando su respuesta, si las siguientes afirmaciones son verdaderas o falsas:

a) La Entrada/Salida por interrupciones es la que proporciona mejor rendimiento en la transferencia de un bloque de datos a un periférico.

Falso. La entrada salida mediante acceso directo a memoria (DMA) es la que proporciona mejor rendimiento en dispositivos de bloque al transmitir la información el dispositivo sin necesidad de intervención de la CPU.

a) El PC es un registro transparente al programador.

Falso. El PC es un registro visible al programador puesto que éste conoce su existencia al estar presente en la especificación del juego de instrucciones del computador. Es un registro de propósito específico puesto que contiene la dirección donde está almacenada la siguiente instrucción a ejecutar.

a) El modelo Von Neumann se basa en etiquetar la información en memoria, es decir, indicar qué zona contiene instrucciones y qué zona contiene datos.

Falso. El modelo Von Neumann se basa en el concepto de programa almacenado: datos e instrucciones residen en una memoria accesible por dirección, pero el contenido de la memoria no está "marcado" o etiquetado como dato o instrucción.

a) El Registro de Instrucción es un registro de propósito específico que contiene la siguiente instrucción a ejecutar.

Falso. El registro de instrucción es un registro transparente que contiene la instrucción que se está ejecutando.

2 Sea un computador de dos direcciones con modelo de ejecución registro-registro con palabra de 32 bits y direccionamiento a nivel de byte que dispone únicamente de direccionamiento inmediato, directo a registro e indirecto a registro. Realice los programas correspondientes a las instrucciones que se muestran a continuación para el computador indicado. Si es necesario utilice los registros RT1, RT2 y RT3 como registros auxiliares.

```
SUB /1000, #4[.R7++]
```

```
MOVE .R7, .RT1
ADD .RT1, #4
LD .RT1, [.RT1]
LD .RT2, #1000
LD .RT3, [.RT2]
SUB .RT3, .RT1
ST .RT3, [.RT2]
ADD .R7, #4
```

```
CALL [#4[.R7]]
```

```
MOVE .R7, .RT1
ADD .RT1, #4
LD .RT2, [.RT1]
CALL [.RT1]
```

```
OR [.R1], [#4[.R7++]]
```

```
MOVE .R7, .RT1
ADD .RT1, #4
LD .RT1, [.RT1]
LD .RT1, [.RT1]
LD .RT2, [.R1]
OR .RT2, .RT1
ST .RT2, [.R1]
ADD .R7, #4
```

```
BR #4[.R7++]
```

```
MOVE .R7, .RT1
ADD .RT1, #4
ADD .R7, #4
BR [.RT1]
```



```

        ld r21, r31, 8           ; leer parámetro texto

        PUSH (r20)               ; pasar parámetro tabla
        PUSH (r21)               ; pasar parámetro texto
        jsr CUENTA               ; llamar a subrutina CUENTA(texto, tabla)
        POP (r21)
        POP (r20)

        or r2, r0, 256
        or r10, r0, r0           ; r10 = cmax

B_CMAX:    ld r3, r20, r0         ; r3 = tabla(i)
           cmp r4, r3, r10
           jbo hi, r4, MENOR      ; si r3 > r10
           or r10, r3, r0         ; cmax = tabla (i)
MENOR:     DJNZ (r2, B_CMAX)

           subu r20, r31, 1024    ; r20 = tabla
           PUSH (r10)             ; pasar parámetro cmax
           PUSH (r20)             ; pasar parámetro tabla
           PUSH (r21)             ; pasar parámetro texto
           jsr RECODIFICA         ; llamar a subrutina RECODIFICA(texto, tabla, cmax)
           POP (r21)
           POP (r20)              ; extraer parámetros de la pila
           POP (r10)

           or r30, r31, r0        ; destruir variables locales
           POP (r31)              ; recuperar fp del llamante
           POP (r1)               ; recuperar dirección de retorno
           jsr (r1)               ; retornar

CUENTA:    PUSH(r1)              ; salvar dirección de retorno
           PUSH (r31)             ; salvar fp del llamante
           or r31, r30, r0        ; crear marco de pila

           ld r20, r31, 8         ; r20 = texto
           ld r21, r31, 12        ; r21 = tabla

B_CUENTA:  ld.ub r2, r20, r0       ; r2 = siguiente carácter
           mulu r3, r2, 4         ; r3 = r2 * 4 => índice de r2 en tabla
           ld r4, r21, r3         ; r4 = frecuencia de r2 en texto
           addu r4, r4, 1         ; incrementar frecuencia de r2
           st r4, r21, r3
           addu r20, r20, 1       ; r20 -> siguiente carácter
           cmp r5, r2, r0
           jbl ne, r5, B_CUENTA  ; si r2 es distinto de null (0x00) -> seguir contando

           or r30, r31, r0        ; destruir variables locales
           POP (r31)              ; recuperar fp del llamante
           POP (r1)               ; recuperar dirección de retorno
           jsr (r1)               ; retornar

```

1 Indique qué tipo de registro es y qué información tiene cada uno de los registros indicados a continuación: Registro de Instrucción, Puntero de pila, Registro de direcciones y Contador de programa.

SOLUCIÓN

- Registro de Instrucción: Es un registro transparente al usuario que contiene la instrucción que se está ejecutando.
- Puntero de Pila: Es un registro de propósito específico que contiene la dirección de la cima de la pila.
- Registro de direcciones: Es un registro transparente al usuario que contiene la dirección de memoria a la que se está accediendo.
- Contador de Programa: Es un registro de propósito específico que contiene la dirección de la siguiente instrucción a ejecutar.

2 Se dispone de un texto almacenado en memoria que finaliza con el carácter NUL (código ASCII 0x00). El texto contiene palabras separadas por un único espacio en blanco (código ASCII 0x20). Se quiere procesar este texto almacenado y detectar las palabras que contengan alguna errata. Para ello, se dispone de una lista de palabras correctas almacenadas en la lista **DICCIONARIO**. Esta lista **DICCIONARIO** es una lista de punteros a palabras correctas formadas por bytes que acaban en un carácter blanco. El fin de la lista es un puntero a NUL 0x00000000.

Se pide programar en ensamblador del MC88110:

a) una subrutina **COMPROBAR_TEXTO** (Texto, Diccionario, Erratas) que compare todas las palabras del texto con las del diccionario, y genere una lista **ERRATAS** con los punteros a las palabras del texto que no estén incluidas en el diccionario. Esta lista finalizará con el valor 0x00000000. (Ver ejemplo de la figura 1.). Los parámetros se pasan por dirección en la pila.

Se supondrá que existe la función de librería **strcmp(str1, str2)** que compara las cadenas de caracteres **str1** y **str2** y devuelve un 0 en **r29** si son iguales o 1 si no lo son. Los parámetros se le pasan por dirección en la pila, y cada cadena de caracteres finaliza con un espacio en blanco (código ASCII 0x20).

b) la subrutina de librería **strcmp(str1, str2)**

Para la realización de este ejercicio se llevará a cabo el tratamiento del Marco de Pila descrito en clase y se supondrá que están definidas todas las macros que se han explicado en la parte teórica de la asignatura; que las subrutinas llamantes dejan disponibles todos los registros excepto **r1**, **r30** (SP) y **r31** (FP); que la pila crece hacia direcciones de memoria decrecientes y el puntero de pila apunta a la última posición ocupada (de la misma forma que se ha utilizado en el proyecto).

SOLUCIÓN

a) Para recorrer el texto se creará una variable local en el marco de pila de la subrutina con la dirección en curso del texto. La subrutina recorre el texto, y para cada palabra recorre la lista del diccionario, llamando a **strcmp** con la entrada en curso del texto y la entrada en curso del diccionario. Previa a cada llamada a **strcmp** se salva la variable local, y después se recupera de la pila. Se continua el recorrido del diccionario hasta que las dos palabras sean iguales o se llegue al final del diccionario. En este último caso hay que insertar la dirección en curso del texto en la entrada correspondiente de la lista de erratas. Si se llega al final del texto hay que completar la lista de erratas insertando un 0.

```
COMPROBAR_TEXTO: PUSH(r1)
                  PUSH(r31)      ; Se crea el marco de pila
                  or r31, r30, r30

                  ld r20, r31, 8   ; leer parametros: r20 = texto
                  ld r21, r31, 12  ; r21 = diccionario
                  ld r22, r31, 16  ; r22 = erratas
```

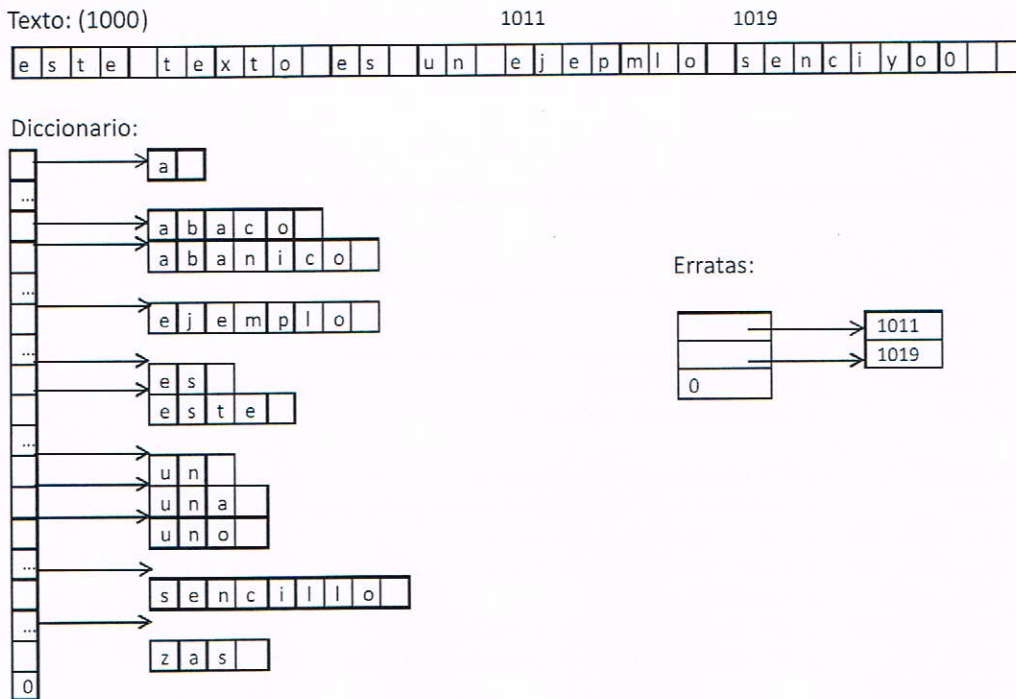



Figura 1

```
or r4, r0, 0x20 ; r4 contiene caracter en blanco
ld r10, r20, 0 ; inic r10 (puntero a pal texto)
```

```
BUCTEX: ld.bu r15, r10, r0
        cmp r5, r15, r0 ; si palabra de texto tiene 0 ir a fin_texto
        bbl eq, r5, FINTEX
        PUSH(r10) ; pasar parametro dir str1 (palabra de texto) en pila
```

```
BUCDIC: ld r11, r21, r0 ; r11 = dir palabra diccionario
        cmp r5, r11, r0 ; si la entrada de diccionario es 0 ir a insertar errata
        bbl eq, r5, ERRATA
        PUSH(r11) ; pasar parametro dir str2 dir (palabra de diccio) en pila
        bsr STRCMP ; comparar las dos palabras
        addu r30, r30, 4 ; eliminar parametro2 de pila
        cmp r5, r29, r0 ; si son distintas las palabras seguir buscando en diccio
```

```
        addu r21, r21, 4 ; avanzar puntero de lista diccionario
        bbl eq, r5, BUCDIC
        ; palabras iguales: ir a siguiente palabra de texto
        addu r30, r30, 4 ; antes, eliminar parametro1 de pila
        ; avanzar puntero de texto a siguiente palabra
```

```
AVANZATEX: ld.bu r15, r10, r0
        cmp r5, r15, r4 ; si car no es blanco seguir avanzando
        addu r10, r10, 1
        bbl ne, r5, AVANZATEX
```

```
        addu r10, r10, 1 ; inicializar puntero a principio de diccio
        br BUCTEX
```

```
ERRATA: st r10, r22, r0
        addu r22, r22, 4
        br AVANZATEX ; ir a avanzar palabra de texto
```

```
FINTEX: st r0, r22, 4 ; insertar fin de lista de erratas
```

```
or r30,r31,r31
POP(r31)
POP(r1)
jmp(r1)
```

b) la subrutina de librería strcmp(str1,str2) puede programarse de la siguiente manera:

```
STRCMP: PUSH(r1)
        PUSH(r31)           ; Se crea el marco de pila
        or r31, r30, r30
        ld r21, r31, 8       ; leer parametros: r21 = dir str1
        ld r22, r31, 12      ; r22 = dir str1
        or r29,r0,r0         ; inicializar r29 a 0

        or r2,r0,0x20        ; r2 contiene el espacio
        or r3,r0,r0          ; inicializa a 0 desplazamiento por string
BUCCMP: ld.bu r11, r21, r3    ; r11= caracter de str1
        ld.bu r12, r21, r3    ; r12= caracter de str2
        cmp r5, r11, r12      ; compara
        bb0 eq, r5, DISTIN    ; si son distintos finaliza
        cmp r5, r11, r2       ; si son iguales y además espacio, fin
        bb1 2, r5, FINCMP
        addu r3, r3, 1         ; si son iguales, incrementar desplazamiento
        br BUCCMP
DISTIN: or r29, r29, 1
FINCMP: POP (r31)
        POP (r1)
        jmp(r1)
```


ESTRUCTURA DE COMPUTADORES

PRIMER PARCIAL (20 de Abril de 2012)

PROBLEMAS

1 (2 puntos) Indique las fases de ejecución de la instrucción de dos palabras `ADD [.R5++], .R4, /1000` (ensamblador IEEE)

SOLUCIÓN

- 1: $PC \rightarrow AR$; Lectura de la segunda palabra de la instrucción
- 2: $M(AR) \rightarrow AR, PC+4 \rightarrow PC$
- 3: $M(AR) \rightarrow TMP1$; Lectura de operando
- 4: $R4+TMP1 \rightarrow DR$; act. EST; Operación y actualización del Reg. Estado
- 5: $R5 \rightarrow AR$; Escritura del resultado
- 6: $DR \rightarrow M(AR)$
- 7: $R5+4 \rightarrow R5$; Postincremento
- 8: $PC \rightarrow AR$; Fetch de la siguiente instrucción
- 9: $M(AR) \rightarrow DR, PC+4 \rightarrow PC$
- 10: $DR \rightarrow IR$, ir a CO
- 11: Decodificación

2 (2,5 puntos) Sea un computador de 32 bits cuyos únicos modos de direccionamiento son: inmediato, directo a registro e indirecto a registro. El modelo de ejecución es registro-registro y la memoria es direccionable a nivel de byte. Indique las secuencias de instrucciones equivalentes a las que se expresan a continuación en ensamblador del IEEE:

SOLUCIÓN

`BR #3[++.R1]`

```
ADD .R1, #4
MOV .R1, .TMP1
ADD .TMP1, #3
BR [.TMP1]
```

`MOV [.R1], #7[.R2--]`

```
MOV .R2, .TMP1
ADD .TMP1, #7
MOV [.R1], [.TMP1]
SUB .R2, #4
```

`ADD .R1, [.R2], #7[--.R4]`

```
SUB .R4, #4
MOV .R4, .TMP1
ADD .TMP1, #7
LD TMP1, [.TMP1]
LD .R1, [.R2]
ADD .R1, .TMP2
```

`DBNZ .R1, #7[.R4++]` (Decrementa el primer operando y si el resultado no es cero salta al segundo operando)

```
MOV .R4, .TMP1
ADD .TMP1, #7
ADD .R4, #4
DBNZ .R1, [.TMP1]
```

3 (6 puntos) Se desea realizar operaciones con matrices dispersas (matrices en las que la mayor parte de sus elementos son nulos). La función `suma` recibe dos parámetros en la pila:

- `p1`: Es una lista compacta que contiene elementos de una matriz dispersa. Se pasa por dirección. Cada uno de los elementos de la lista se compone de tres campos enteros:
 - `elemento`: Contiene el elemento que pertenece a la matriz.
 - `fila`: Indica la fila a la que pertenece el elemento. Las filas comienzan a numerarse en la posición 0.

- **columna:** Indica la columna a la que pertenece el elemento. Las columnas comienzan a numerarse en la posición 0.

El final de la lista está indicado con el el valor -1 contenido en el campo **fila**.

- **p2:** Es otra lista compacta que contiene los elementos de otra matriz dispersa almacenados de la misma forma que la matriz **p1**. Se pasa por dirección.

Esta función realizará la suma de las dos matrices representadas por **p1** y **p2** almacenando el resultado en **p1**.

Para facilitar la construcción de la subrutina anterior, la función de librería **elemento(p, fila, columna)** devuelve en el registro **r29** la dirección que ocupa el elemento indicado por los dos últimos parámetros (**fila** y **columna**) en la matriz dispersa **p** que se pasan en la pila. Si el elemento no existe en la matriz representada por **p**, se devuelve en **r29** el valor **0xffffffff**.

Programa las siguientes subrutinas:

- a) La función **anade(p, valor, fila, columna)** que añade el elemento descrito en los tres últimos parámetros a la matriz indicada en **p**.
- b) La función **suma(p1, p2)** descrita al inicio del enunciado utilizando las funciones **elemento** y **anade**.

El resultado del ejemplo que se muestra en la figura será la suma de las matrices dispersas **P1** y **P2**. Los elementos (1,5) y (0,5) serán el resultado de la suma de los elementos (1,5) y (0,5) en las dos matrices. El elemento (0,0) solo está contenido en **P2** y por tanto dicho elemento será añadido a **P1** (matriz resultado).

P1	20	1	5	15	0	5	40	4	8	5	-1	-1			
P2	5	0	0	15	1	5	10	0	5	5	-1	-1			
P1 final	35	1	5	25	0	5	40	4	8	5	0	0	4	-1	-1

SOLUCIÓN

a) La función **anade** busca el valor -1 en el campo **fila** de cada uno de los elementos de la lista que se pasa como primer parámetro. Cuando se encuentra, se inserta el elemento el sustitución del elemento finalizador de la lista incluyendo el valor, **fila** y **columna** que se pasan como parámetros. A continuación del nuevo elemento se inserta un elemento finalizador de la lista que contiene el valor -1 en el campo **fila**. A continuación se muestra la función **anade**:

```

anade:  ld r20,r30,r0          ; Carga p
bucle:  ld r2,r20,4
        cmp r5,r2,-1
        bbl eq, r5, fin_buc
        addu r20,r20,12       ; No se ha encontrado el final y
        br bucle              ; se avanza una posición en la lista.
fin_buc:ld r2,r30,4           ; Se carga y almacena valor
        st r2,r20,r0
        ld r2,r30,8           ; Se carga y almacena fila
        st r2,r20,4
        ld r2,r30,12          ; Se carga y almacena columna
        st r2,r20,8
        add r2,r0,-1           ; Se genera una última entrada en que la fila y
        st r2,r20,16          ; columna contienen -1
        st r2,r20,20
        jmp(r1)

```

b) La función **suma** recorre la lista **p2**. Cada elemento de la lista **p2** se busca en la lista **p1** (llamado a la función **elemento** con el valor de **fila** y **columna** del elemento de **p2**). Si el elemento no está contenido en **p1**, se añade como nuevo elemento invocando a la función **anade**. Si no está contenido se suma el valor del elemento de **p2** al valor del elemento de **p1**. Si el resultado de esta suma es 0, el elemento es nulo, no debe estar en la lista **p1** y se compacta la lista eliminando el elemento nulo. Para realizar esta operación se ha generado una variable local en el marco de pila que contiene el puntero en curso a la lista **p2**. A continuación se muestra la función **suma**:


```

suma: PUSH(r1)
      PUSH(r31)
      or r31,r30,r30
      subu r30,r30,4      ; puntero a p2
      ld r20,r31,12
bucle_s: ld r2,r20,4      ; Si la fila del elemento de p2 es -1
      cmp r5,r2,-1      ; se acaba
      bbl eq,r5,final
      ld r3,r20,8      ; Se llama a elemento para buscar el elemento de p2 en p1
      st r20,r31,-4
      PUSH(r3)
      PUSH(r2)
      ld r20,r31,8
      PUSH(r20)
      bsr elemento
      addu r30,r30,12    ; Se eliminan los parámetros de la pila
      cmp r5,r29,-1
      bbl eq,r5,no_existe
      ld r20,r31,-4      ; Se recupera el puntero a p2
      ld r2,r29,r0
      ld r3,r20,r0      ; El elemento existe. Se suman los dos valores de los
      add r2,r2,r3      ; elementos de p1 y p2
      addu r20,r20,12    ; Se avanza al siguiente elemento de p2
      cmp r5,r2,r0
      bbl eq,r5,compacta
      st r2,r29,r0      ; Se almacena el resultado de la suma en p1
      br bucle_s
compacta: ld r2,r29,12    ; Si el resultado de la suma es 0, se elimina el elemento
      st r2,r29,r0      ; de la lista compactándola
      ld r2,r29,16
      st r2,r29,4
      ld r3,r29,20
      st r3,r29,8
      cmp r5,r3,-1
      bbl eq,r5,bucle_s
      addu r29,r29,12
      br compacta
no_existe: ld r21,r31,8 ; Si el elemento no existe se añade invocando a anade
      ld r20,r31,-4
      ld r2,r20,8
      PUSH(r2)          ; columna
      ld r2,r20,4
      PUSH(r2)          ; fila
      ld r2,r20,r0
      PUSH(r2)          ; valor
      PUSH(r21)         ; p1
      bsr anade
      addu r30,r30,8
      ld r20,r31,-4      ; Se recupera el puntero a p2 y se avanza al siguiente
      addu r20,r20,12
      br bucle_s
final: or r30,r31,r31
      POP(r31)
      POP(r1)
      jmp(r1)

```

1 (2 puntos) Indique, de forma razonada, cuáles de las siguientes afirmaciones son verdaderas y cuáles falsas:

SOLUCIÓN

- El registro PC tiene la dirección de la instrucción que se está ejecutando
Falso. El PC contiene la dirección de la siguiente instrucción a la que se está ejecutando.
- Los registros transparentes al usuario se pueden utilizar para pasar parámetros a subrutinas
Falso. Los registros transparentes no se pueden direccionar por los usuarios como operandos.
- La Unidad de Control necesita como entrada solo señales internas de la CPU: reloj, registro de estado, etc...
Falso. La UC necesita también las señales de E/S.
- El registro contador de programa puede contener tanto direcciones como datos si hace falta usarlo como registro auxiliar.
Falso. El PC sólo puede tener direcciones, en concreto la dirección de la siguiente instrucción.

2 (8 puntos) Se dispone de una lista de palabras almacenadas en la lista *Diccionario* ordenadas alfabéticamente en orden creciente. Esta lista *Diccionario* es una lista de punteros a palabras formadas por bytes que acaban en un carácter NUL (`\0`) y la lista contiene palabras que comienzan por todas las letras (desde la 'a' hasta la 'z'). El fin de la lista es un puntero a *NULL* `0x00000000`. El código ASCII de la letra 'a' es `0x61` y el de la 'z' es `0x7a`.

Se pide programar en ensamblador del MC88110:

a) una subrutina *BUSCAR* (*Indice*, *Palabra*) que busca la palabra *Palabra* en el diccionario que se pasa en el primer parámetro como la tabla *Indice* de 26 entradas. La subrutina devuelve en *r29* un 0 si la palabra está en el diccionario o `0xffffffff` si no está.

Se supondrá que existe la función de librería `strcmp(str1, str2)` que compara las cadenas de caracteres *str1* y *str2* y devuelve un 0 en *r29* si son iguales o 1 si no lo son. Los parámetros se le pasan por dirección en la pila, y cada cadena de caracteres finaliza con el carácter NUL (código ASCII `0x00`).

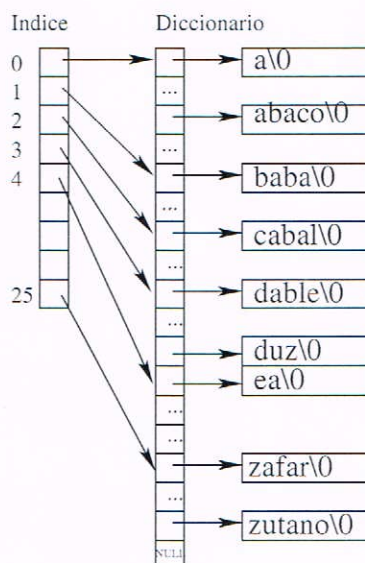


Figura 1

Para la realización de este ejercicio se llevará a cabo el tratamiento del Marco de Pila descrito en clase y se supondrá que están definidas todas las macros que se han explicado en la parte teórica de la asignatura; que las subrutinas llamantes dejan disponibles todos los registros excepto *r1*, *r30* (SP) y *r31* (FP); que la pila crece hacia direcciones de memoria decrecientes y el puntero de pila apunta a la última posición ocupada (de la misma forma que se ha utilizado en el proyecto).

SOLUCIÓN

La solución crea el marco de pila para contener dos variables locales. El puntero a la tabla de diccionario que ocupa el desplazamiento -4 al puntero de marco r31 y la letra de comienzo de la palabra que ocupa la posición -1. La búsqueda sigue los siguientes pasos:

- Se obtiene la palabra y su primera letra. Al valor ASCII de esta letra se le resta el valor 0x61 para obtener la entrada del índice correspondiente a la letra.
- Se multiplica por 4 para obtener el desplazamiento de la entrada que hay que obtener.
- Se obtiene la dirección del diccionario donde están almacenadas las palabras que comienzan por dicha letra.
- Se realiza una búsqueda de la palabra hasta que las letras de comienzo de la palabra parámetro y la que se encuentra en el diccionario no son iguales o se encuentra el final de la tabla. En estos casos se sale de la función con el valor 1. Si se encuentra se sale con el valor 0 (mismo valor que devuelve strcmp).

```

BUSCAR: PUSH(r1)
        PUSH(r31)
        or r31,r30,r30
        subu r30,r30,8
        ld r20,r31,12      ; Se carga la palabra
        ld.bu r5,r20,r0     ; Se carga la primera letra
        st r5,r31,-8       ; Se almacena en una var. local
        subu r4,r5,0x61     ; Se calcula la entrada
        mulu r4,r4,4        ; Se calcula el desplazamiento al comienzo del índice
        ld r21,r31,8       ; Índice
        ld r21,r21,r4       ; Puntero al una entrada del diccionario
BUCLE:  st r21,r31,-4       ; Se almacena en una var. local
        PUSH(r20)          ; Se pasa la palabra
        ld r21,r21,r0       ; Palabra del diccionario
        PUSH(r21)          ; Se pasa la palabra del diccionario
        bsr strcmp
        addu r30,r30,8
        cmp r4,r29,r0
        bbl eq,r4,fin      ; Si se encuentra se acaba
        ld r21,r31,-4
        ld r20,r31,12
        ld r5,r31,-8
        addu r21,r21,4
        ld r22,r21,r0
        cmp r4,r22,r0
        bbl eq,r4,fin_1    ; Si hemos llegado al final se sale con -1
        ld.bu r4,r22,r0    ; Se carga la primera letra de la palabra del diccionario
        cmp r4,r4,r5       ; Si las letras de comienzo no son iguale se sale con -1
        bbl ne,r4,fin_1
        br BUCLE
fin_1:  subu r29,r0,1
fin:    or r30,r31,r31
        POP(r31)
        POP(r1)
        jmp(r1)

```

NOTAS: 27 de junio de 2012

REVISIÓN: 29 de junio de 2012

DURACIÓN: 1 hora 30 minutos

PUNTUACIÓN: Especificada en cada problema

1 (1,5 puntos) En el diseño de la estructura de un computador se consideran los siguientes tiempos:

- lectura/escritura de registro: 1 ut
- lectura o escritura del banco de registros: 5 ut
- retardo de la ALU: 35 ut
- decodificador: 4 ut
- buffer triestado: despreciable
- multiplexores: 2 ut
- acceso a Memoria de Control: 20 ut
- acceso a Memoria Principal: 100 ut
- secuenciador de microprograma: 8 ut

Calcule el mínimo tiempo del ciclo de reloj del computador, suponiendo que su unidad de control es:

- a) cableada
- b) microprogramada

SOLUCIÓN

a) cableada

El ciclo de reloj debe ser suficiente para poder ejecutar la operación elemental de mayor duración.

$$T_{reloj} = t_{OP} = t_{BR} + t_{MUX} + t_{ALU} + t_{BR} = 5 + 2 + 35 + 5 = 47 \text{ ut}$$

b) microprogramada

La duración del ciclo de reloj debe dar tiempo a realizar la operación elemental más larga teniendo en cuenta que se actualice el estado, opere el secuenciador y lea en la memoria de control:

$$T_{reloj} = t_{RC} + t_{dec} + t_{BR} + t_{MUX} + t_{ALU} + t_{SR} + t_{SR} + t_{SEC} + t_{RD} + t_{RD} + t_{MC} + t_{RC} = 1 + 4 + 5 + 2 + 35 + 1 + 1 + 8 + 1 + 1 + 20 + 1 = 80 \text{ ut}$$

En cualquier caso, la operación de memoria principal necesitaría de varios ciclos de reloj (3 ó 2) por ser esta memoria externa a la CPU.

2 (1,5 puntos) Exprese el mayor y el menor de los números representables en los siguientes formatos numéricos de 8 bits y calcule su valor decimal, indicando también cuales son las respectivas representaciones del cero. Obtenga el valor decimal de las secuencias de bits $X = 11110011$ e $Y = 00111000$ en cada uno de los formatos.

a) Enteros en exceso a 128.

b) Coma flotante en signo-magnitud con 1 bit de signo, 4 bits de exponente en exceso 8 y 3 bits de mantisa con bit implícito y la coma a la izquierda de éste.

SOLUCIÓN

a)

$$N^{\circ} \text{ mayor: } 11111111 = 255 - 128 = 127$$

$$N^{\circ} \text{ menor: } 00000000 = 0 - 128 = -128$$

$$\text{Representación del cero: } 10000000 = 0 + 128$$

$$X = 243 - 128 = 115$$

$$Y = 56 - 128 = -72$$

b)

$$N^{\circ} \text{ mayor: } 01111111 = ,1111.2^7 = 1111000 = 120$$

$$N^{\circ} \text{ menor: } 11111111 = -,1111.2^7 = -1111000 = -120$$

Puesto que se trata de una mantisa normalizada con bit implícito, para expresar el cero se reservan las siguientes representaciones:

Representaciones del cero: 00000000 y 10000000

$$X = -,1011.2^6 = -101100 = -44$$

$$Y = ,1000.2^{-1} = ,01 = 0,25$$

3 (3,5 puntos) Un computador representa números de coma flotante con un formato de 16 bits que sigue las convenciones del estándar IEEE754 en todo excepto en los tamaños, es decir, usa el mismo tipo de representaciones especiales, representación del exponente, bit implícito, situación de la coma, representaciones normalizadas y no normalizadas, así como bits de guarda y redondeo. En el formato, el bit superior corresponde al signo, los seis siguientes al exponente y los nueve últimos a la magnitud de la mantisa.

- a) Determine el rango de representación y la resolución del formato.
- b) Determine el valor decimal de los siguientes números que están representados en el formato:

$$A = H'4805 \quad B = H'BC04 \quad C = H'802C \quad D = H'7E00$$

c) Realice paso a paso la suma $A + B$ utilizando redondeo al más próximo. Exprese el resultado en el formato de partida y determine su valor decimal.

SOLUCIÓN

1. Rango y resolución del formato.

Números normalizados.

Exponente: El estándar IEEE754 representa los exponentes en exceso a $2^{n-1} - 1$. En este caso sería exceso a 31. Como se reserva el exponente mínimo (-31) para la representación del cero y los números no normalizados, y el exponente máximo ($+32$) para la representación del infinito y las indeterminaciones, el rango de exponente para la representación de números queda: $[-30, 31]$.

La mantisas normalizadas en este formato se representan en signo-magnitud, con bit implícito y la coma situada a la derecha del bit implícito:

$$\text{Mantisa: } \pm \begin{cases} 1,000000000 & \rightarrow 1 \\ 1,111111111 & \rightarrow 2 - 2^{-9} \end{cases}$$

$$\text{El rango para números normalizados es: } \pm [1 \cdot 2^{-30}, (2 - 2^{-9}) \cdot 2^{31}]$$

Números no normalizados

Exponente: Siguiendo el estándar IEEE754, aunque los números no normalizados se representan con el exponente todo a ceros (valor -31), todos ellos tienen como exponente el más pequeño de los normalizados, en este caso -30 . De este modo hay continuidad en la representación.

$$\text{Mantisas: } \pm \begin{cases} 0,000000000 & \rightarrow 0 \\ 0,000000001 & \rightarrow 2^{-9} \\ 0,111111111 & \rightarrow 1 - 2^{-9} \end{cases}$$

$$\text{El rango para números no normalizados es: } \pm [2^{-9} \cdot 2^{-30}, (1 - 2^{-9}) \cdot 2^{-30}] \cup 0$$

Así pues, el rango total es:

$$\pm [1 \cdot 2^{-30}, (2 - 2^{-9}) \cdot 2^{31}] \cup \pm [2^{-9} \cdot 2^{-30}, (1 - 2^{-9}) \cdot 2^{-30}] \cup 0$$

La resolución depende del exponente y es: $2^{-9} \cdot 2^E$

2. Valor decimal de los números.

$$A = H'4805 = 0 \ 100100 \ 000000101 = +1,000000101 \cdot 2^5 = +100000,0101 = +32,3125$$

$$B = H'BC04 = 1 \ 011110 \ 000000100 = -1,000000100 \cdot 2^{-1} = -0,1000000100 = -0,50390625$$

$$C = H'802C = 1 \ 000000 \ 000101100$$

Como se puede ver, el exponente del número C es cero, con lo cual representa un número no normalizado, al que se le asigna el exponente mínimo (-30).

$$C = -0,00101100 \cdot 2^{-30} = -1011 \cdot 2^{-37} = -8 \cdot 10^{-11}$$

$$D = H'7E00 = 0 \ 111111 \ 000000000 = +\infty$$

3. Suma $A + B$.

Este formato utiliza dos bits de guarda y un bit retenedor para la suma. Los números a sumar son:

$$A = +1,000000101 \cdot 2^5 \text{ y } B = -1,000000100 \cdot 2^{-1}$$

Se restan los exponentes: $E_A - E_B = 5 - (-1) = 6$. Hay que desplazar la mantisa de B seis lugares a la derecha. Así, teniendo en cuenta los dos bits de guarda y el bit retenedor queda:

$$B = -0,000001000 \ 00 \ 1 \cdot 2^5$$

M_A	1,000000101 00 0
M_B	- 0,000001000 00 1
	0,111111100 11 1 $\cdot 2^5$
Normalizacion	1,111111001 11 0 $\cdot 2^4$
Redondeo	0,000000000 10 0
	1,111111010 01 0 $\cdot 2^4$

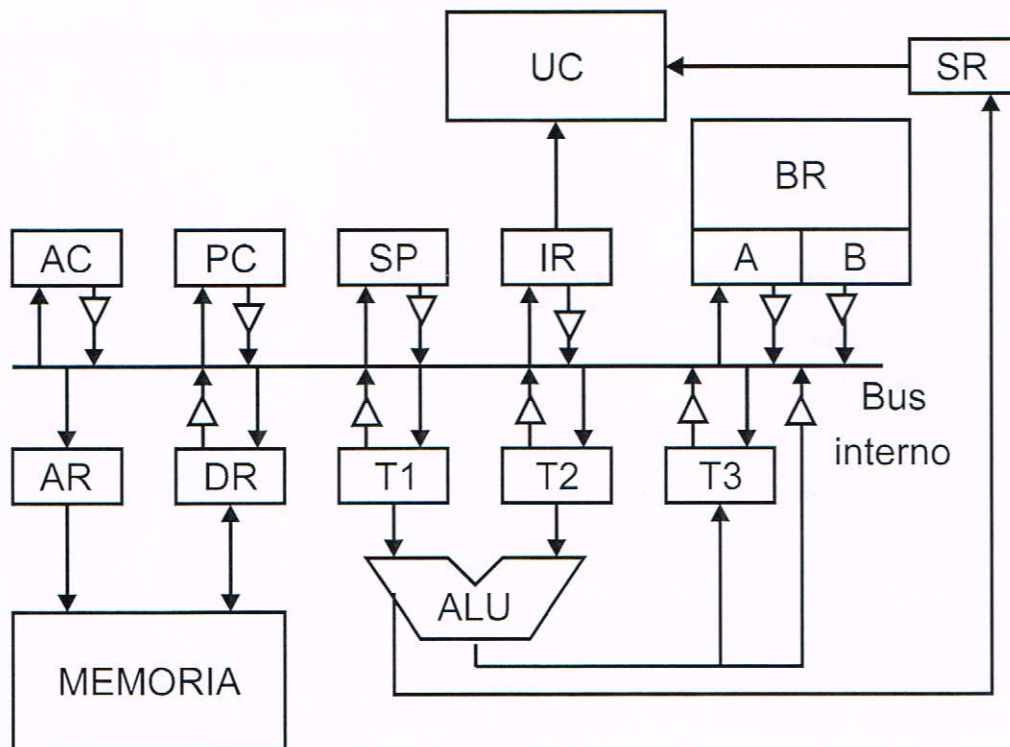
Representación:

$$A + B = +1,111111010 \cdot 2^4 = 0 \ 1000111 \ 11111010 = \text{H}'47\text{FA}$$

Valor decimal:

$$A + B = +1,111111010 \cdot 2^4 = +11111,1101 = 31,8125$$

4 (3,5 puntos) En la figura se muestra el esquema de un computador de 64 bits con Unidad de control microprogramada y direccionamiento a nivel de byte. Los accesos a memoria tienen una duración media de 2 ciclos de reloj. La pila crece hacia direcciones decrecientes y el puntero de pila (SP) apunta a la primera posición vacía de la cima de la pila. Los incrementos o decrementos de los registros, se realizan a través de la ALU.



a) Realice, a nivel RT (transferencia entre registros), el microprograma de la microsubrutina de fetch

b) Realice a nivel RT el microprograma de la fase de ejecución de la instrucción de una palabra, perteneciente al juego de instrucciones de este computador: CALLNZ #desp[++.R2]. Esta instrucción realiza un salto a subrutina condicional si NZ.

c) De acuerdo con los microprogramas de los apartados anteriores y considerando una frecuencia de reloj de 500 MHz, determine el tiempo medio que tarda en ejecutarse la instrucción del apartado b.

SOLUCIÓN

a)

La microsubrutina de fetch puede realizarse con cuatro ciclos:

```
f1: T1, AR ← PC
f2: Acceso a memoria, PC ← T1 + 8
f3: DR ← M(AR)
f4: IR ← DR, microsalto a C.O.
```

b)

El microprograma de la fase de ejecución sería:

```
m1: Si Z microsalto a fetch
m2: T1, AR ← SP
m3: DR ← PC
m4: Acceso a memoria, SP ← T1 - 8
m5: M(AR) ← DR, T2 ← R2
m6: T2, R2 ← T2 + 8
m7: T1 ← IR(desp)
m8: PC ← T2 + T1, microsalto a fetch
```

c)

$$T_{ck} = 1/500MHz = 2ns$$

$$\text{Si } Z = 0$$

$$t_{NZ_{ejecucion}} = T_{ck} \cdot n^{\circ}ciclos = (2 \cdot (4 + 8))ns = 24ns$$

$$\text{Si } Z = 1$$

$$t_{Z_{ejecucion}} = T_{ck} \cdot n^{\circ}ciclos = (2 \cdot (4 + 1))ns = 10ns$$

1 Un computador cuenta con los dos formatos de representación siguientes:

Formato 1. Coma fija, 12 bits en complemento a dos y con la coma situada en el centro: P. ej. 011011,100101

Formato 2. Coma flotante, 12 bits, el bit superior es el signo, los 5 siguientes el exponente representado en exceso a 16 y los 6 siguientes la mantisa, representada en signo magnitud, con bit implícito y la coma situada a la derecha del bit implícito.

a) Determine el rango y resolución de ambos formatos.

b) Represente en ambos formatos los números $A=26,35$ y $B=-0,41$.

c) Realice la suma $A+B$ en ambos formatos determinando el valor decimal del resultado. Para el formato de coma flotante utilice un bit de guarda y un bit retenedor y redondeo por forzado a 1.

SOLUCIÓN

a) Rango y resolución de los formatos.

Formato 1.

$$\begin{cases} 100000,000000 \rightarrow -2^5 \\ 011111,111111 \rightarrow 2^5 - 2^{-6} \end{cases}$$

$$\text{Rango} = [-2^5, 2^5 - 2^{-6}]$$

$$\text{Resolución} = 2^{-6}$$

Formato 2.

Exponente: $[-16, 15]$. Como hay que reservar el exponente -16 para la representación del cero, el rango del exponente queda: $[-15, 15]$.

$$\text{Mantisa: } \pm \begin{cases} 1,000000 \rightarrow 1 \\ 1,111111 \rightarrow 2 - 2^{-6} \end{cases}$$

$$\text{Rango} = \pm [1 \cdot 2^{-15}, (2 - 2^{-6}) \cdot 2^{15}] \cup 0$$

$$\text{Resolución} = 2^{-6} \cdot 2^E$$

b) Representación de los números

$$A = H'440B = 0 \ 100010 \ 000001011 = +1,000001011 \cdot 2^2 = 100,0001011 = 4,0859375$$

$$B = H'BDDA = 1 \ 011110 \ 111011010 = -1,111011010 \cdot 2^{-2} = -,011110110100 = -0,481445312$$

$$A = +26,35 = H'+1A,59 = +11010,01011001$$

$$B = -0,41 = H'-,68F = -,01101000111$$

Formato 1.

$$A = 011010,010110 = H'696$$

$$B = -000000,011010 = 111111,100110 = H'FE6$$

Formato 2.

$$A = +1,101001 \cdot 2^4 = 0 \ 10100 \ 101001 = H'529$$

$$B = +1,101000 \cdot 2^{-2} = 1 \ 01110 \ 101000 = H'BA8$$

c) Suma $A+B$.

Formato 1.

$$\begin{array}{r} A \qquad \qquad \qquad 011010,010110 \\ +B \qquad \qquad \qquad 111111,100110 \\ \hline A+B \qquad \qquad \qquad 011001,111100 \end{array}$$

$$A+B = 011001,111100 = H'67C = +25,9375$$

Seguidamente, carga el registro R3 con este valor.

Las operaciones elementales que se realizan en cada ciclo de reloj, incluyendo la fase de fetch, son las siguiente:

```
c1: AR<- PC
c2: ICM, R, PC<- PC + 1
c3: DR<- M(AR)
c4: RI<- DR
c5: Decodificación
c6: AR<- PC
c7: ICM, R, PC<- PC + 1
c8: DR<- M(AR)
c9: AR<- DR
c10: ICM, R
c11: DR<- M(AR)
c12: R3<- DR
```

b) El cronograma de la fase de ejecución de la instrucción, se muestra en hoja aparte.

Figura 2. Plantilla del cronograma a entregar

Apellidos, Nombre..... SOLUCIÓN b) N° de Matrícula.....

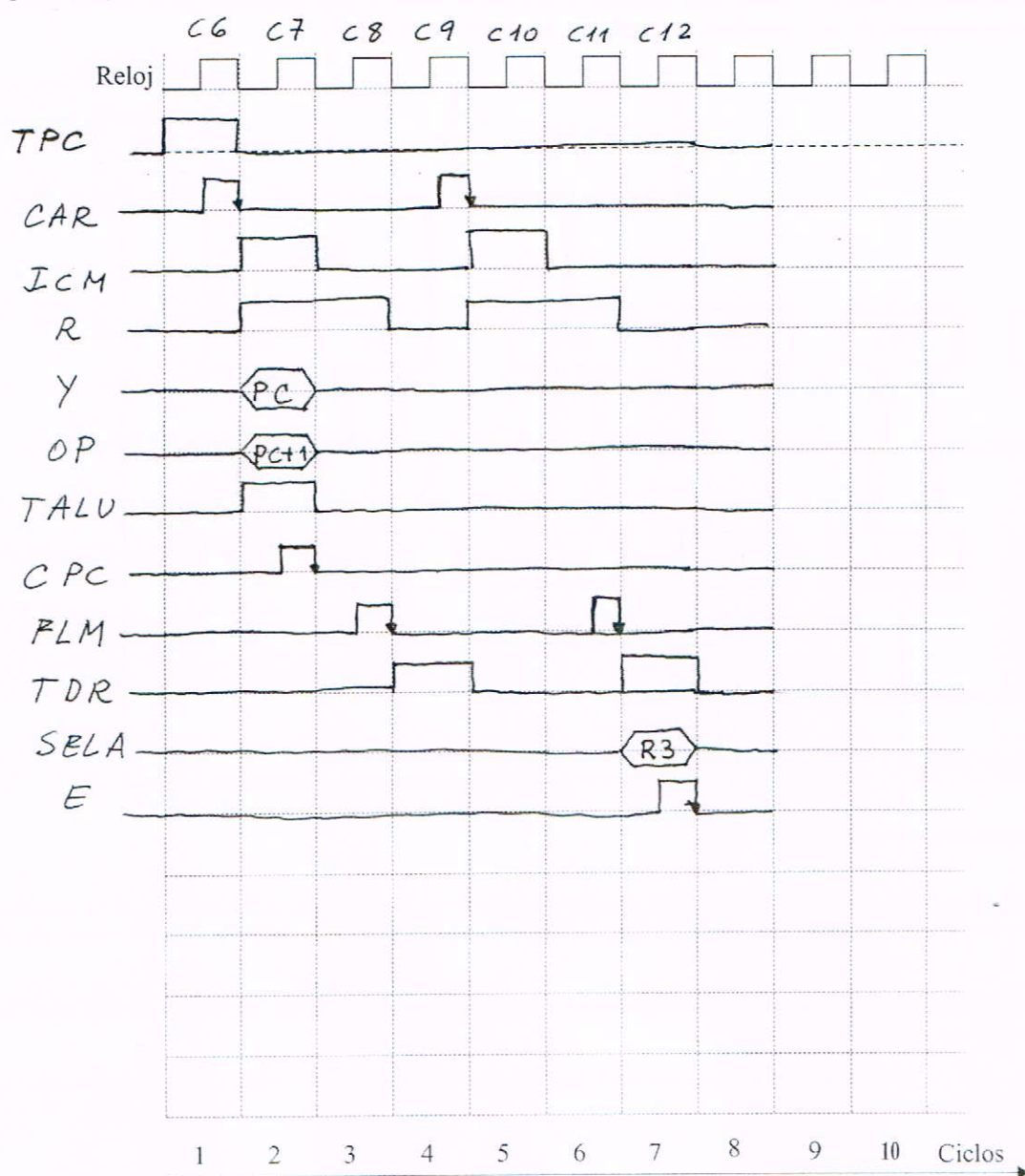
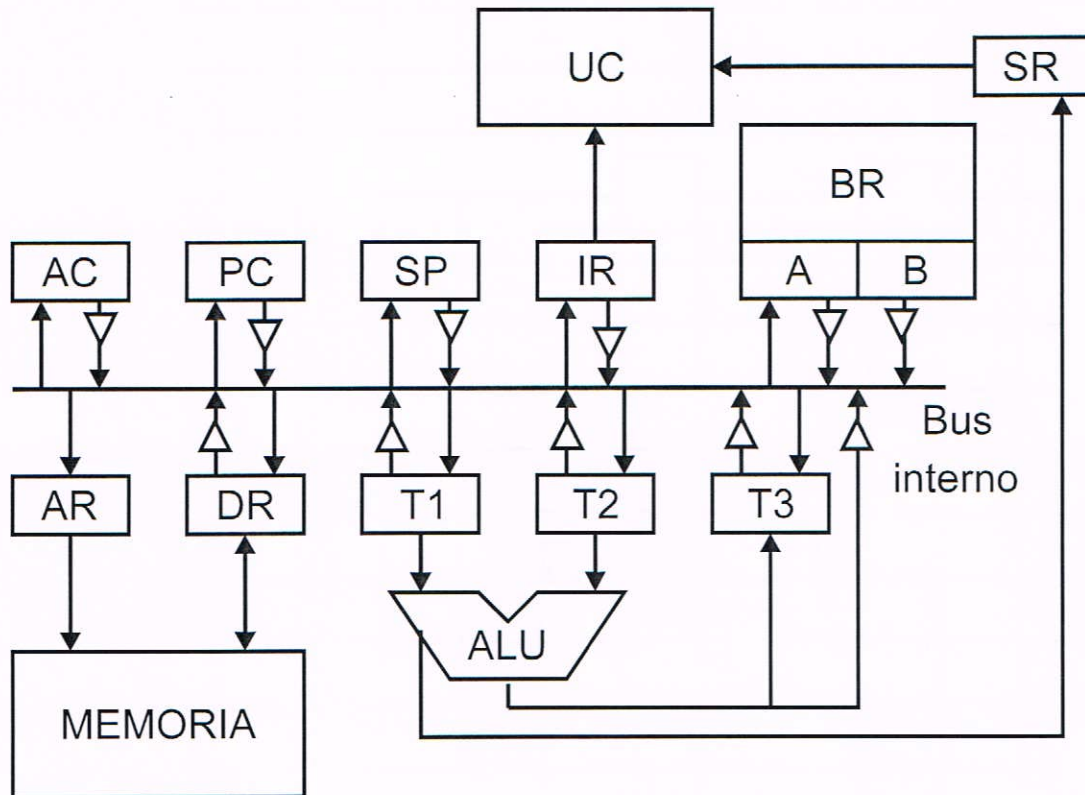


Figura 2. Plantilla del cronograma a entregar

1 (5 puntos) En la figura se muestra el esquema de un computador de 32 bits con Unidad de control microprogramada y direccionamiento a nivel de byte. Los accesos a memoria tienen una duración de 2 ciclos de reloj. Los incrementos o decrementos de los registros, se realizan a través de la ALU.



a) Determine el número de accesos a memoria, tanto para lectura como para escritura, que se producen durante la ejecución de la instrucción de una palabra: `SUB [.R1], [.R2--]`, que realiza la resta de dos operandos.

b) Exprese a nivel RT (transferencia entre registros) el microprograma de la instrucción del apartado anterior. Incluya la microsubrutina de fetch.

c) Considerando que el procesador trabaja con una frecuencia de reloj de 2 GHz, calcule el tiempo que tarda en ejecutarse la instrucción.

SOLUCIÓN

a)

- Fase de fetch: Se produce un acceso a memoria para leer la instrucción.
- Fase de ejecución: Se producen dos accesos a memoria para leer los operandos y un acceso de escritura del resultado.
- El número total de accesos a memoria son cuatro, tres de lectura y uno de escritura.

b) El microprograma correspondiente a la instrucción es el siguiente:

Fetch:

```

m1: AR, T1<- PC
m2: Acceso a memoria, PC<- T1 + 4
m3: DR<- M(AR)
m4: IR<- DR, microsalto a C. 0.

```


Fase de ejecución:

```

m5: T1, AR ← R2
m6: Acceso a memoria, R2 ← T1 - 4
m7: DR ← M(AR)
m8: AR ← R1
m9: Acceso a memoria, T2 ← DR
m10: DR ← M(AR)
m11: T1 ← DR
m12: DR ← T1 - T2; Actualizar SR
m13: Acceso a memoria
m14: M(AR) ← DR, microsalto a fetch

```

c)

$$T_{ck} = 1/f_{ck} = 0,5 \cdot 10^{-9} s = 0,5 ns$$

$$te_{jec} = T_{ck} \cdot n^{\circ} \text{ciclos} = (0,5 \cdot 14) ns = 7 ns$$

2 (5 puntos) Se tiene un formato de coma flotante de 16 bits cuyo bit superior representa el bit de signo, los siete siguientes el exponente expresado en exceso a 64 y los ocho bits siguientes la magnitud de la mantisa, con bit implícito y la coma situada a la derecha de éste.

a) Determine el rango de representación y la resolución del formato.

b) Dados los números: $A = H'C702$ y $B = H'45F6$ expresados en el formato arriba indicado, calcule justificadamente sus valores decimales.

c) Realice la operación $A - B$, en coma flotante, utilizando 2 bits de guarda y redondeo al más próximo. Expresé el resultado en el formato y obtenga su valor decimal.

SOLUCIÓN

a) Rango y resolución del formato.

Exponente: $[-64, 63]$. Reservando el exponente -64 para la representación del cero, el rango del exponente queda: $[-63, 63]$.

$$\text{Mantisa: } \pm \begin{cases} 1,00000000 \rightarrow 1 \\ 1,11111111 \rightarrow 2 - 2^{-8} \end{cases}$$

$$\text{Rango} = \pm [1 \cdot 2^{-63}, (2 - 2^{-8}) \cdot 2^{63}] \cup 0$$

$$\text{Resolución} = 2^{-8} \cdot 2^E$$

b) Valor decimal y representación de los números A y B en el formato.

$$A = 1100011100000010 = -1,00000010 \cdot 2^7 = -10000001,0 = -129$$

$$B = 0100010111101110 = 1,11101110 \cdot 2^5 = 111110,110 = -62,75$$

c) Operación $A - B$.

$$\text{Los números a restar son: } A = -1,00000010 \cdot 2^7 \text{ y } B = 1,11101110 \cdot 2^5$$

Los exponentes son distintos: $E_A > E_B$, por lo que el exponente del resultado (a falta de posibles postnormalizaciones) es $E = E_A = 7$

Se restan los exponentes: $E_A - E_B = 2$, lo cual indica que hay desplazar la mantisa de B dos lugares a la derecha, usando dos bits de guarda:

$$B = -0,01111101 \ 10 \cdot 2^7$$

$$\text{La operación se realiza como: } A - B = A + (-B)$$

Puesto que los signos de A y - B son iguales, hay que sumar el valor absoluto de las mantisas y poner el signo común.

1,00000010	00	
+ 0,01111101	10	
1,01111111	10	
+	1	
1,10000000		

normalizado.

redondeo al más próximo

normalizado

$$A - B = -1,10000000 \cdot 2^7 = -11000000,0 = -192.$$

En el formato:

$$A - B = 1 \ 1000111 \ 10000000$$

1 (5 puntos) La Figura 1 muestra el diagrama de la estructura del procesador de un computador de 32 bits cuya unidad de control es cableada. Los accesos a memoria tienen una duración de 1 ciclo de reloj y el direccionamiento es a nivel de palabra. La pila crece hacia direcciones de memoria decrecientes. Como puntero de pila se utiliza el registro R1 del banco de registros que apunta a la primera posición libre en la cima de la pila. Los incrementos o decrementos de cualquier registro del computador, se realizan a través de la ALU.

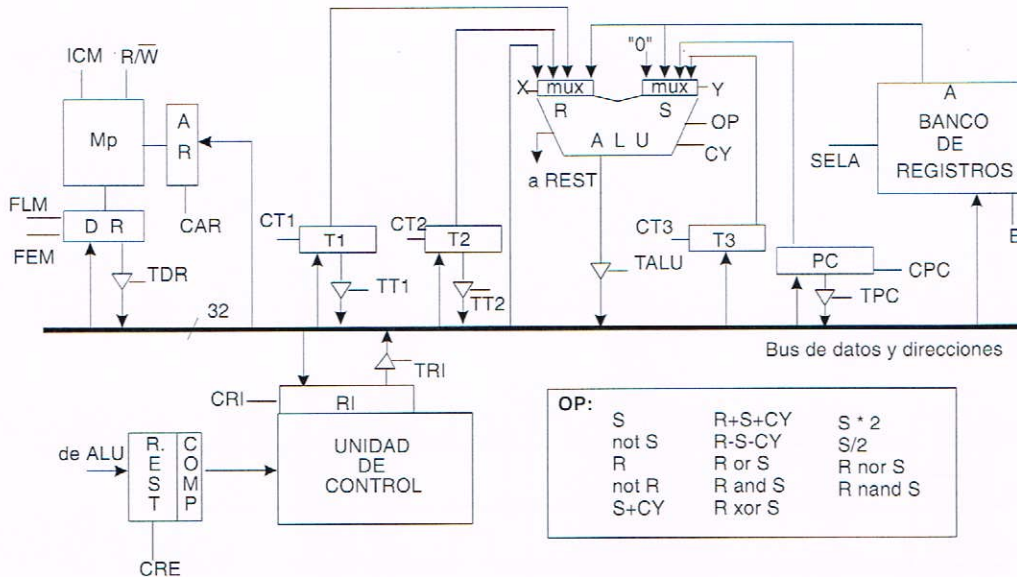


Figura 1. Estructura de la CPU y operaciones de la ALU

a) Exprese a nivel RT (transferencia entre registros), en esta misma hoja, las operaciones elementales que se producen en cada uno de los ciclos de la ejecución de la instrucción de una palabra POP [.R3] .

SOLUCIÓN

Las operaciones elementales realizadas en este computador para cada ciclo de reloj, durante la ejecución de la instrucción anterior, quedan especificadas como sigue:

Fetch

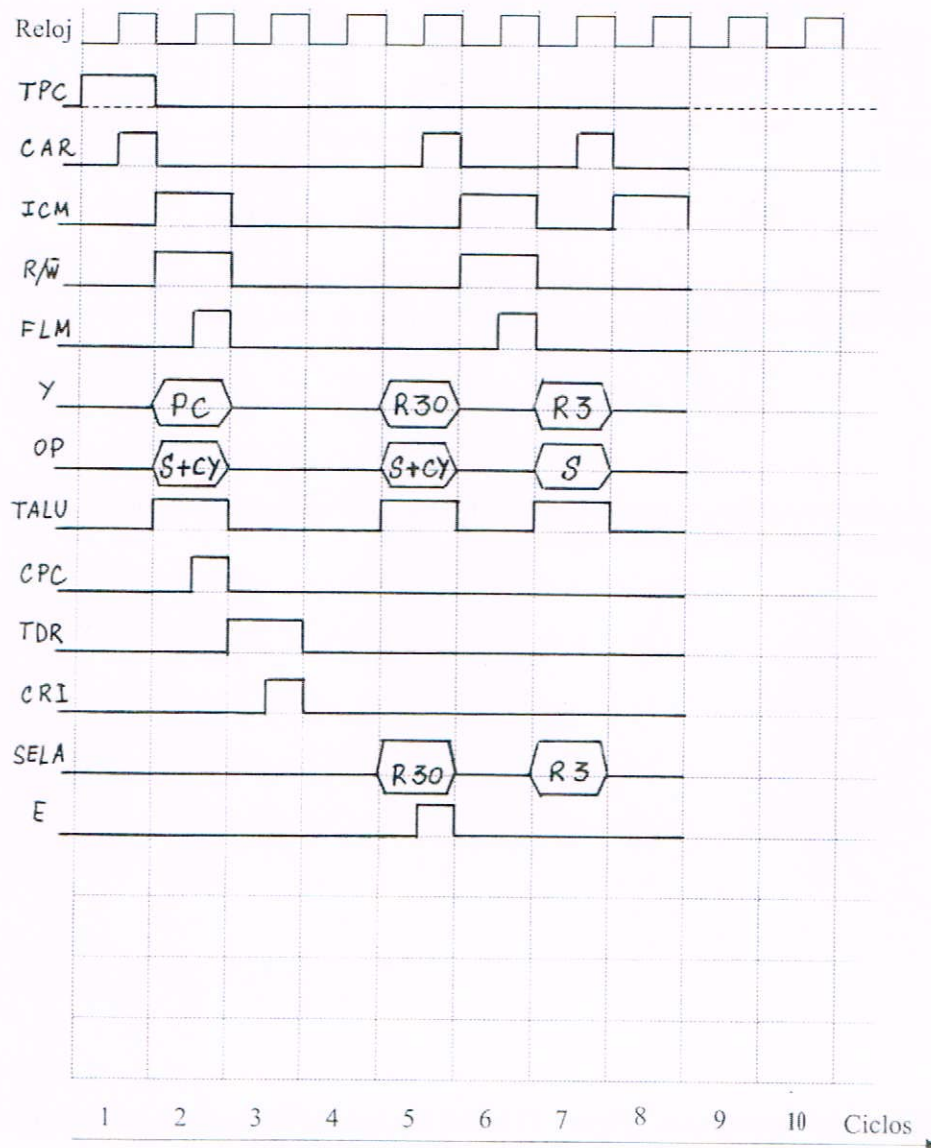
- 1: AR ← PC
- 2: DR ← M(AR), PC ← PC + 1
- 3: RI ← DR
- 4: Decodificación

Fase de ejecución

- 5: AR, R30 ← R30 + 1
- 6: DR ← M(AR)
- 7: AR ← R3
- 8: M(AR) ← DR

a) Represente en la plantilla del cronograma (Figura 2) todas las señales de control que se activan en cada uno de los ciclos de reloj correspondientes a la ejecución de la instrucción anterior.

Figura 2. Cronograma de las señales de control activadas durante la ejecución de la instrucción



2 (5 puntos) La ALU de un computador opera con un formato de coma flotante de 16 bits con mantisa en signo-magnitud y exponente en exceso 32. El primer bit corresponde al signo, los seis siguientes al exponente y los nueve últimos a la magnitud de la mantisa. Dispone de bit implícito con la coma a la izquierda de éste. El operador aritmético utiliza redondeo al más próximo, dos bits de guarda y un bit retenedor.

a) Determine el rango y la resolución del formato anterior.

b) Dado el número $A = H'EE567$ representado en coma fija y complemento a 1, con 14 bits de parte entera y 6 bits de parte fraccionaria, obtenga su valor decimal y expréselo en el formato de coma flotante.

c) Considerando el número $B = H'183C$ representado en coma fija y signo magnitud, con el primer bit para el signo, 11 bits de parte entera y 4 bits de parte fraccionaria, obtenga su valor decimal y expréselo en el formato de coma flotante.

d) Realice razonadamente la operación $A + B$ en el formato de coma flotante y obtenga el valor decimal del resultado.

SOLUCIÓN

a) Rango y resolución del formato en coma flotante.

Exponente: $[-32, 31]$. Como hay que reservar el exponente -32 para la representación del cero, el rango del exponente queda: $[-31, 31]$.

$$\text{Mantisa: } \pm \begin{cases} ,1000000000 & \rightarrow 2^{-1} \\ ,1111111111 & \rightarrow 1 - 2^{-10} \end{cases}$$

$$\text{Rango} = \pm [-2^{-1} \cdot 2^{-31}, (1 - 2^{-10}) \cdot 2^{31}] \cup 0$$

$$\text{Resolución} = 2^{-10} \cdot 2^E$$

b) Valor decimal del número A y representación en el formato de coma flotante.

$$A = H'EE567 = 11101110010101,100111 = (\text{complementando a 1}) -00010001101010,011000 = -1130,375$$

Aplicando redondeo al más próximo para representar A en el formato de coma flotante:

$$A = -,1000110101 \cdot 2^{11} = 1 \ 101011 \ 000110101 = H'D635$$

c) Valor decimal del número B y representación en el formato de coma flotante.

$$B = H'183C = 000110000011,1100 = 387,75$$

Aplicando redondeo al más próximo para representar B en el formato de coma flotante:

$$B = ,1100001000 \cdot 2^9 = 0 \ 101001 \ 100001000 = H'5308$$

d) Suma $A + B$.

Los números a sumar son: $A = -,1000110101 \cdot 2^{11}$ y $B = ,1100001000 \cdot 2^9$

Como los exponentes son distintos, el exponente del resultado es: $E = E_A = 11$.

Se restan los exponentes: $E_A - E_B = 2$.

Hay que desplazar la mantisa de B dos lugares a la derecha, utilizando dos bits de guarda y un bit retenedor:

$$B = -,0011000010 \ 00 \ 0 \cdot 2^{11}$$

Puesto que los signos de A y B son distintos, deben compararse las mantisas tras haber igualado los exponentes al de A.

Ya que la mantisa de A es mayor que la de B, la operación a realizar es $MA - MB$, siendo el signo del resultado $S = SA = 1$.

$$\begin{array}{r}
 ,1000110101 \quad 00 \quad 0 \\
 - ,0011000010 \quad 00 \\
 \hline
 ,0101110011 \quad 00 \quad 0 \\
 ,1011100110 \quad 00 \\
 + \quad \quad \quad 1 \\
 \hline
 ,1011100110
 \end{array}$$

No normalizado.

Normalización. Decremento del exponente: $E = 10$

Redondeo

$$A + B = -,1011100110 \cdot 2^{10} = -1011100110 = -742.$$

En el formato:

$$A + B = 1 \ 101010 \ 011100110 = H'D4E6$$

1 (1 punto) Sea una cinta magnética de 1 TB capacidad, con una densidad de grabación lineal de 16.000 bits/mm. Calcule cuántos bloques de 1 MB bytes puede almacenar si los claros IRC son de 2 cm de longitud.

SOLUCIÓN

$$\frac{10^{12}}{10^6 + (16.000 \times 20/8)} = 998.001 \text{ bloques}$$

2 (1 punto) Dado un monitor LCD con una resolución de 1.920×1.200 píxeles, 32 bits de profundidad de color y 60 Hz de frecuencia de refresco,

a) indique si puede ser utilizado conectándolo a una placa gráfica cuya memoria de pantalla es de 64 bits y tiene un tiempo de acceso de 10 ns.

b) Calcule cuál es la máxima frecuencia de refresco a la que puede funcionar el monitor con esa placa gráfica.

SOLUCIÓN

a) Requisitos del monitor:

$$1.920 \times 1.200 \times 32/8 \times 60 = 552.960.000 \text{ bytes/s}$$

Capacidad de la placa gráfica:

$$\frac{64/8}{10 \times 10^{-9}} = 800.000.000 \text{ bytes/s}$$

$800.000.000 \text{ bytes/s} \geq 552.960.000 \text{ bytes/s}$: El monitor puede usarse con la placa gráfica.

b) Máxima frecuencia de refresco:

$$\frac{800.000.000 \text{ bytes/s}}{1.920 \times 1.200 \times 32/8 \text{ bytes}} = 86,80 \text{ Hz}$$

3 (1 punto) Calcule cuánto tiempo tarda en transmitirse un bloque de 9.600 bytes por una línea serie asíncrona de 57.600 bits/s, 2 bits de stop y bit de paridad par en los casos siguientes:

a) Si la conexión es half-duplex.

b) Si la conexión es full-duplex.

SOLUCIÓN

a) Se transmiten 12 bits por cada byte (start, dato, paridad y stop):

$$\frac{9.600 \times 12 \text{ bits}}{57.600 \text{ bits/s}} = 2 \text{ s.}$$

b) Se transmite la misma información a la misma velocidad: 2 s.

4 Sea una unidad de disco duro de brazo móvil con las siguientes características:

- $81,92 \cdot 10^9$ bytes de capacidad bruta y 80 % de almacenamiento neto.
- 16 superficies, 200 sectores por pista y 512 bytes de información neta por sector.
- Velocidad de rotación: 12.000 rpm.
- El tiempo que emplea en mover la cabeza de un cilindro a otro consecutivo es 0,01 ms.
- El tiempo de estabilización de las cabezas es 1 ms.

a) (2 puntos) Calcule el número de cilindros y la velocidad de transferencia de la unidad de disco.